指针进阶

- 指针

```c
int a = 6;
int *p = &a;//指针，是存放地址的变量。取地址运算 &
*p = 999;// 通过指针变量里存的地址值可以间接访问a变量
int a[2] = {1,2}, *p=a;
char c[]="hello", *q=c;
p=p+1; //p里面的地址值偏移一个int
q=q+1; //q里面的地址值偏移一个char
p-a; // 相差的int个数  1
q-c; // 相差的char个数 1
p+a; // 错误
q+c; // 错误
printf("%d", *p++);  printf("%d", *++p);
printf("%d", (*p)++) printf("%d", ++*p);
```

- 一维数组

```c
int main()
{   // a是一个数组，每一个成员是一个int数据
    int a[] = {1,2,3};
    fun1(a, 3);
}
void fun1(int a[], int n)
{  //等价于int *a  传数组实际上传的是数组首地址，也就是指针
    for (int i = 0; i < n; i++)
        printf("%d\n", a[i]);
}
```

- 字符串

```c
// 字符串常量，实际上就是这串字符的首地址，也就是指针
char *p = "hello,world";
printf("%s", p);
```

- 指针数组：一个数组，里面每个成员都是一个指针。

```c
#include<stdio.h>
int main()
{   // t是一个数组，每一个成员是一个指针
    char *t[] = {"computer", "phone", "mp4"};
    for (int i = 0; i < 3; i++)
        printf("%s\n", t[i]);
}
```

- 指向指针的指针

```c
int a = 3;
int *pa = &a;  // 指向int的指针
int **p = &pa; // 指向指针的指针
printf("%d", **p);
```

```c
//例：指针数组与指向指针的指针
int main()
{   // t是一个数组，每一个成员是一个指针
    char *t[] = {"computer", "phone", "mp4"};
    fun2(t, 3);
}
void fun(char *t[], int n)
{ // 传数组实际是传首地址也就是指针,指针指向的内容里每一个也是一个指针,所以t是指向指针的指针,char **t
    for (int i = 0; i < n; i++)
        printf("%s\n", t[i]);
}
```

```c
//例：主函数原型。举例：程序名 命令行参数 ping 192.168.1.100
int main (int argc, char *argv[])
{
    for (int i = 0; i < argc; i++)
        printf("%s\n", argv[i]);
}
```

- 下面程序运行结果是（　　）

```c
#include<stdio.h>
int main()
{   char *a[] = {"dog", "cat", "chook"};
    char **p = a;
    printf("%s,%c", *(p+2), *(*(p+1) + 2))
}
```

- （2009）下面程序运行结果是（　　）

```c
int main()
{   char **p, *t[] = {"computer", "phone", "mp4"};
    for (p = t+2; p >= t; p--)
        printf("%c", *(*p+1));
}
```

- （2011高考）以下函数功能是：在指针数组表示的字符串列表中查找特定的字符，指针数组以`NULL`指针结束，如果找到返回`TRUE`，否则返回`FALSE`，程序中有两处错误，不得删行或增行。

```c
#include<stdio.h>
#include<string.h>
#define TRUE 1
#define FALSE 0
int find_string(char **strings, char value)
{   char *cur_str;
    if (*strings == NULL)
        return FALSE;
    while ( (cur_str=*string) != NULL)
```

```
    {    while (cur_str != '\0')
            if (*cur_str++ == value)
                return TRUE;
    }
    return FALSE;
}
```

- 二维数组

```
#include<stdio.h>
int main()
{int a[3][3] = {{1,2,3},{4,5,6},{7,8,9}}, i, j;
    for (i = 0; i < 3; i++){
        for (j = 0; j < 3; j++)
            printf("%3d", a[i][j]);
        printf("\n");
    }
}
```

- 指向数组的指针：指针存的是地址，这个地址位置存放的是一个数组。二维数组，实际上是一个一维数组，里面的每个元素也是一个一维数组。二维数组在作为函数形参时，传的是数组的首地址，这个地址位置存放的也是一个数组，因此用指向数组的指针。

```
#include<stdio.h>
int main()
{   int a[3][3] = {{1,2,3},{4,5,6},{7,8,9}}, i, j;
    int (*p)[3]; // 指向数组的指针
    p = a; //p的列数3  要和数组a的一致
    for (i = 0; i < 3; i++){
        for (j = 0; j < 3; j++)
            printf("%3d", *(*(p+i)+j) );
        printf("\n");
    }
}
```

- （2011高考）下面程序输出结果是（　　）

```
#include<stdio.h>
int main()
{   char a[] = {'a', 'b', 'c', 'd'};
    char *p = (char *)(&a + 1);
    printf("%c,%c", *(a+1), *(p-1));
}
```

- （2011高考）下面程序`fun()`函数的功能是：从N个字符串中找出最长的那个串，并将其地址作为函数返回值。

```
#include<stdio.h>
#include<string.h>
#define N 4
#define M 50
char *fun(char (*q)[M])
{   int i; char *p;
    ____________
    for (i = 0; i < N; i++)
```

```
        if (strlen(p) < _________ )
            _______________
    return p;
}
main()
{
    char str[N][M] = {"pingpong", "basketball",
                      "field hockey", "softball"};
    char *longest = fun(str);
    printf("The longest string: %s\n", longest);
}
//注意：如果改为指针数组，函数形参应改为指向指针的指针
//char *str[] = {"pingpong", "basketball", "field hockey", "softball"};
//char *fun(char *q[]){  // 不能写成 char (*q)[M]
//}
```

- （2019年高考题）若有定义语句`int a[3][5];`，按内存中的数据存放顺序，a数组的第10个元素是（　　）

```
A. a[1][4]  B. a[1][3]   C. a[2][3]   D. a[2][4]
```

- （2012高考）C语言中一个2行3列的矩阵M如下所示，得到数值5的表达式是（　　）

```
3   8   9
2   5   6

A. *(M+1)+1          B. *(*M+1)+1
C. *(*(M+1)+1)       D. **((*M+1)+1)
```

- （2020年高考题）若`int a[2][3]={1,2,3,4,5,6};`则不能访问`a[1][2]`的是（　）

```
A. *(a[1]+2)     B. *(*(a+1)+2)
C. *(a+2)        D. *(&a[1][0]+2)
```

- （2015年高考题）下列程序的运行结果是（　　）

```
main()
{
    int a[3][3]={{0,1},{2,3},{4,5}},i,j,s=0;
    for(i=1;i<3;i++)
        for(j=0;j<=i;j++)
            s+=a[i][j];
    printf("%d",s);
}
```

- (2012年高考)以下程序将数组中元素逆序输出，运行结果如下。程序中存在2处错误。

```
dd   cc   bb   aa
```

```
L1    #include<stdio.h>
L2    main()
L3    {
L4        char  *array[]={"aa", "bb", "cc", "dd"};
L5        char   (**pt)[] ;
L6        int  j;
L7        pt=array+3;
L8        for(j=3; j>=0; j--)
L9            printf("%s  ", *(pt)[j]);
L10   }
```

- （2013年高考题）矩阵乘法是指两个矩阵相乘，生成一个新矩阵，其乘法公式如下。以下程序实现了两个2×2的矩阵相乘，只对程序中的2行代码进行修改，使其可以正确运行。

$$\begin{bmatrix} a_{11} & \cdots & a_{1k} \\ \cdots & \cdots & \cdots \\ a_{m1} & \cdots & a_{mk} \end{bmatrix} \times \begin{bmatrix} b_{11} & \cdots & b_{1n} \\ \cdots & \cdots & \cdots \\ b_{k1} & \cdots & b_{kn} \end{bmatrix} = \begin{bmatrix} c_{11} & \cdots & c_{1n} \\ \cdots & \cdots & \cdots \\ c_{m1} & \cdots & c_{mn} \end{bmatrix}$$

$$\text{其中} c_{ij} = \sum_{l=1}^{k} a_{il} \times b_{lj}, i \in [1,m], j \in [1,n]$$

```
L1 #include <stdio.h>
L2 #define M 2
L3 #define N 2
L4 #define K 2
L5 int main(){
L6    int a[M][K]={1,-1,0,2};
L7    int b[K][N]={4,0,2,5};
L8    int c[M][N];
L9    int i,j,l;
L10   for(i=0;i<M;i++){
L11       for(j=0;j<N;j++){
L12           for(l=0;l<K;l++){
L13               c[i][j]=a[i][l]*b[l][j];
L14           }
L15       }
L16   }
L17   return 0;
L18 }
```

- 函数指针

```
int add(int, int); //函数申明
int add(int a, int b){ //函数定义
    return a+b;
}
int axb(int a, int b){
    return a*b;
}
void output(){
    printf("函数指针：就是存放函数内存地址的变量。\n");
}
int main(){
    int (*pf)(int, int); //函数指针d定义
    pf = add; //函数指针赋值，函数名就是函数的地址
```

```
    int c = pf(100,200);//等价于add(100,200)
                //也可以写成(*pf)(100,200)
    pf = axb; //函数指针赋值
    c = pf(100,200);//等价于axb(100,200);
    void (*pf)(); //函数指针定义
    pf(); // 等价于 output();
}
```

函数指针作为函数形参

```
void sort(int a[], int n, int (*compare)(int, int))
{
    int i, j;
    for (i = 0; i < n-1; i++)
        for (j = i+1; j < n; j++)
            if (compare(a[i], a[j]))
            {
                int t = a[i];a[i]=a[j];a[j]=t;
            }
}
int increasing(int a, int b)
{
    return a > b;
}
int descending(int a, int b)
{
    return a < b;
}
void output(int a[], int n){
    int i;
    for (i = 0; i < 10; i++)
        printf("%4d", a[i]);
    printf("\n");
}
int main()
{
    int a[10], i;
    for (i = 0; i < 10; i++)
        a[i] = rand()%100;
    sort(a,10,increasing);
    output(a,10);
    sort(a,10,descending);
    output(a,10);
}
```

小结

- 指针：内存地址
- 数组：`int score[10]; char name[64];`
- 字符串：`char *str = "The C Programing Language";`
- 二维数组：`int a[3][3]={{1,2,3},{4,5,6},{7,8,9}};`
- 指针数组：`char *s[3]={"C++","Java", "Python"};`
- 指向指针的指针：`char **p; p = s;`
- 指向数组的指针：`int (*p)[3]; p = a;`
- 函数指针：函数的内存地址。