

Welcome

数据科学与大数据技术专业 程序设计基础(C语言)

上海体育学院经济管理学院

Wu Ying

一、单项选择题（共15题，每题2分，共计30分）

二、判断题（共5题，每题1分，共计5分）

三、填空题（共8空，每空3分，共计24分）

四、读程序写结果（共3题，每题4分，共计12分）

五、编程题（共2题，共计29分）



通过指针引用字符串

字符串的引用方式

通过字符指针变量输出一个字符串

```
#include <stdio.h>
int main()
{
    char *ps="I love China!";    //定义字符指针变量并初始化
    printf("%s\n",string);        //输出字符串
    return 0;
}
```

```
int main()
{
    char* ps = "Qingyuanhuan Road 650";
    printf("%s\n", ps);

    ps = "Yangpu District, Shanghai.";
    printf("%s\n", ps);

    return 0;
}
```



```
Microsoft Visual Studio 调试控制台
Qingyuanhuan Road 650
Yangpu District, Shanghai.
C:\Users\HP\Desktop\CPrac
按任意键关闭此窗口. . .
```

ps

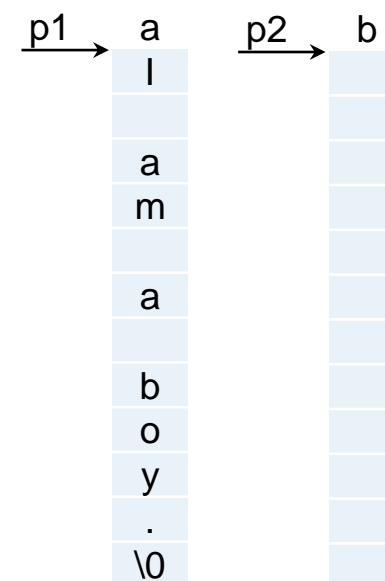
第1个字符
的地址

I
l
o
v
e

C
h
i
n
a
!
\0

字符串的引用方式

【例8.19】用指针变量将字符串a复制给字符串b，然后输出字符串b。



字符串的引用方式

【例8.19】用指针变量将字符串a复制给字符串b，然后输出字符串b。改为函数。

```
int main()
{
    char a[] = "Qingyuanhuan Road 650";
    char b[50];
    char* p1, * p2;
    p1 = a; p2 = b;
    while (*p1 != '\0')
        *p2++ = *p1++;
    *p2 = '\0';

    printf("字符串a: %s\n", a);
    printf("字符串b: %s\n", b);

    return 0;
}
```

Microsoft Visual Studio 调试控制台

```
字符串a: Qingyuanhuan Road 650
字符串b: Qingyuanhuan Road 650
```

```
void mystrcpy(char* sr, char* dt)
{
    while (*sr != '\0')
        *dt++ = *sr++;
    *dt = '\0';
}
```

动态分配内存

用malloc函数开辟动态存储区

函数原型为 `void *malloc(unsigned int size);`

作用是在内存的动态存储区中分配一个长度为size的连续空间。

返回值是所分配区域的第一个字节的地址——此函数是一个指针型函数

```
malloc(100);           //开辟100字节的临时分配域，函数值为其第1个字节的地址
```

指针的基类型为void，即不指向任何类型的数据，只提供一个纯地址。

如果此函数未能成功地执行(例如内存空间不足)，则返回空指针(NULL)。

建立动态一维数组

```
1  #define _CRT_SECURE_NO_WARNINGS
2  #include <stdio.h>
3  #include <stdlib.h>
4
5  int main()
6  {
7      int n;
8      int* p, *h;
9      int i;
10     while (1)
11     {
12         printf("请输入数组大小: \n");
13         scanf("%d", &n);
14         if (n == 0)
15             break;
16         h = (int*)malloc(n * sizeof(int));
17         p = h;
18         printf("\n");
19         printf("请输入 %d 个数组元素: \n", n);
20
21         for (i = 0; i < n; i++)
22             scanf("%d", p + i);
23
24         printf("\n");
25         printf("该数组为: \n");
26
27         p = h;
28         for (i = 0; i < n; i++)
29             printf("%d\t", *(p + i));
30
31         printf("\n\n");
32         free(h);
33     }
34     return 0;
```

Microsoft Visual Studio 调试控制台

请输入数组大小:

3

请输入 3 个数组元素:

11 22 33

该数组为:

11 22 33

请输入数组大小:

10

请输入 10 个数组元素:

0 1 2 3 4 5 6 7 8 9

该数组为:

0 1 2 3 4 5 6 7 8 9

请输入数组大小:

0

C:\Users\HP\Desktop\CPractice\Temp\Debug\Temp.exe (进程 20552) 已退出, 代码为 0。

按任意键关闭此窗口. . .

指向指针的指针变量

指向指针数据的指针变量，简称为指向指针的指针。

p的前面有两个*号。p指向一个 int 指针变量（这个 int 指针指向一个 int 型数据）

```
int **p;
```

如果引用*p，就得到p所指向的 int 指针变量的值。

```
*p
```

```
int a = 6;  
int** pp;  
int* p;  
p = &a;  
pp = &p;  
printf("%d\n", *p);  
printf("%p\n", *pp);
```

A screenshot of a Microsoft Windows command prompt window. The title bar says "C:\ Microsoft". The command prompt shows the output of the printf statements: "6" on the first line and "008FFE44" on the second line.

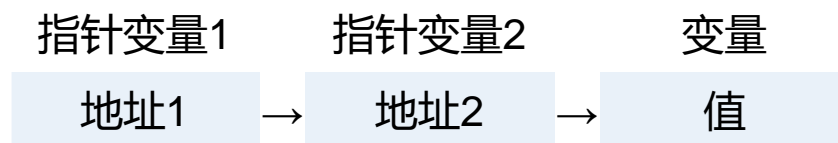
C:\ Microsoft
6
008FFE44

指向指针的指针变量

如果在一个指针变量中存放一个目标变量的地址，这就是“单级间址”；



指向指针数据的指针用的是“二级间址”方法；



取内容——“解引用”

利用二级指针动态建立二维数组

```

1  #define _CRT_SECURE_NO_WARNINGS
2  #include <stdio.h>
3  #include <stdlib.h>
4  #include <string.h>
5  #include <time.h>
6  int** MallocArray(int row, int col);
7  void DeleteArray(int** p, int row);
8  void ShowArray(int** p, int row, int col);
9  int main(void)
10 {
11     int nr, nc; //行数, 列数
12     int** p;
13     nr = 5;
14     nc = 4;
15
16     p = MallocArray(nr, nc);
17     ShowArray(p, nr, nc);
18     DeleteArray(p, nc);
19
20     return 0;
21 }

```

```

22 int** MallocArray(int row, int col)
23 {
24     int i, j;
25     int** pa;
26     srand((unsigned)time(NULL));
27     pa = (int**)malloc(row * sizeof(int*));
28     for (i = 0; i < row; i++)
29     {
30         *(pa + i) = (int*)malloc(col * sizeof(int));
31     }
32
33     for (i = 0; i < row; i++)
34     {
35         for (j = 0; j < col; j++)
36         {
37             (*(pa + i) + j) = rand() % 21;
38         }
39     }
40
41     return pa;
42 }

```

```

43 void DeleteArray(int** p, int row)
44 {
45     int i;
46     for (i = 0; i < row; i++)
47         free(p[i]);
48     free(p);
49 }
50 void ShowArray(int** p, int row, int col)
51 {
52     int i, j;
53     for (i = 0; i < row; i++)
54     {
55         for (j = 0; j < col; j++)
56         {
57             printf("%d\t", *(p + i) + j);
58         }
59         printf("\n");
60     }
61 }
62 }

```

先申请一维的指针数组，然后该数组中的每个指针再申请数组，相当于二维数组

但是这种方法会导致每行可能不相邻，从而访问效率比较低。

Microsoft Visual Studio 调试控制台

```

4      17      12      13
19      5       2      14
9       0      10      6
11      18      16      19
5       12      8       20

```

返回指针值的函数

类型名 *函数名(参数表列)

一个函数可以返回一个整型值、字符值、实型值等，也可以返回指针型的数据

```
int *fun(int x,int y);
```