

# 第8章 指针

——指针变量的定义、初始化及解引用



---

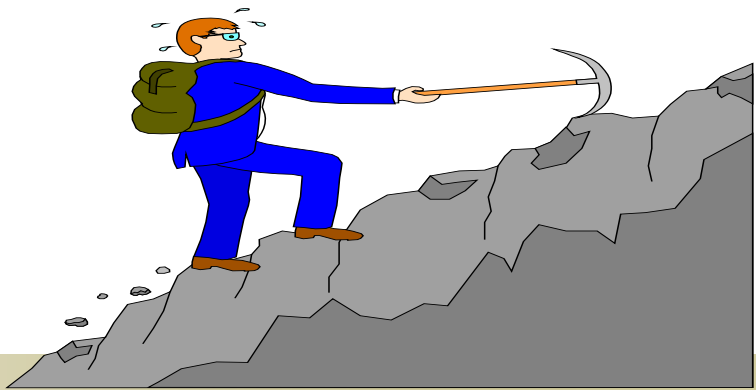
# 本节要讨论的主要问题

- 变量的寻址方式哪有几种？
- 何为指针？如何定义指针类型的变量？
- 如何访问指针变量指向的存储单元中的数据？



# 指针

- 指针（Pointer）是“稀饭（C Fans）”最挚爱的武器
- C的高效、高能主要来自于指针
- 很多“Mission Impossible”由指针完成
  - \* 大多数语言都有无数的“不可能”
  - \* 而C语言是
    - “一切皆有可能” —— 
    - “Impossible is Nothing” —— 



# 指针

强转与指针，并称C语言的两大神器  
用好了可呼风唤雨，威力无比  
用不好也会伤及自身



屠  
龙  
刀

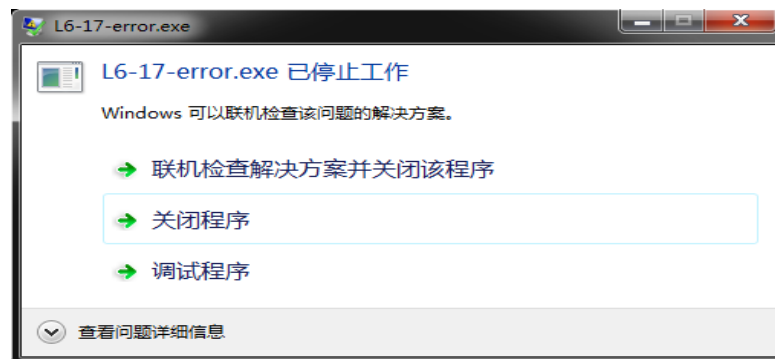


倚  
天  
剑



# 指针

- 是谁惹的祸？
  - 几乎全是由指针和数组引起的非法内存访问导致的
- 黑客攻击服务器利用的bug绝大部分都是指针和数组造成的



- 理解指针要从变量的地址谈起

# 内存如何编址？

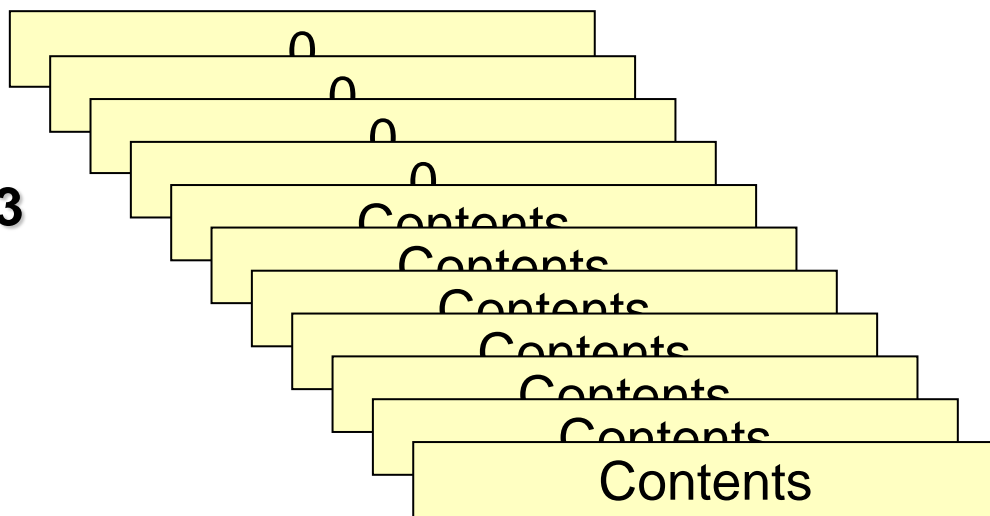
内存中的每个字节都有唯一的编号（地址）  
内存地址按字节编号，其字长一般与主机相同  
32位机使用32位地址，最多支持 $2^{32}$ 字节内存(4G)

0x0037b000

0x0037b001

0x0037b002

0x0037b003



某存储区域

内存

0x00000000

0x00000001

0x00000002

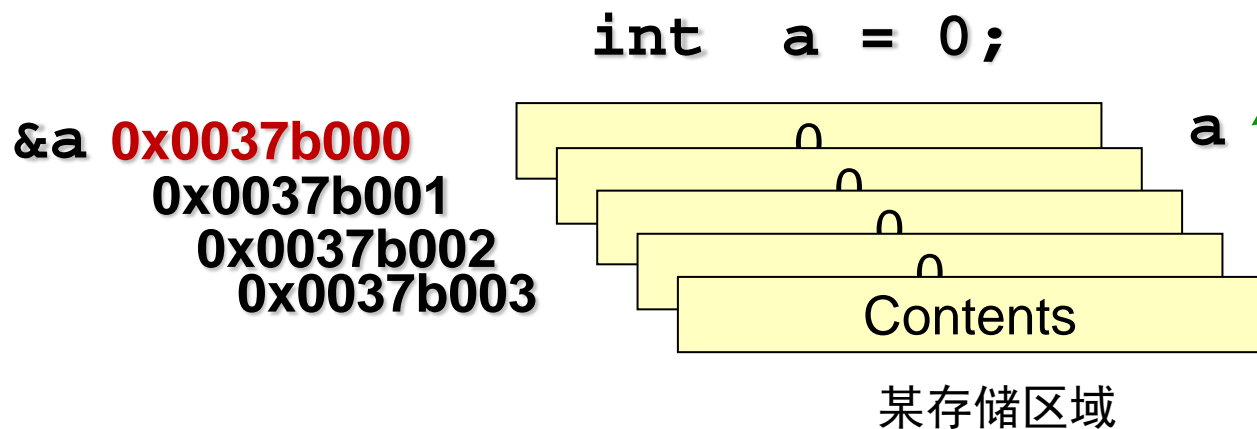
⋮

0xFFFFFFFF

4G内存

地址是一个**无符号整数**，从0开始，依次递增。在表达和交流时，通常把地址写成**十六进制数**

# 如何对变量进行寻址？



直接到变量名标识的存储单元中  
读取变量的值——直接寻址

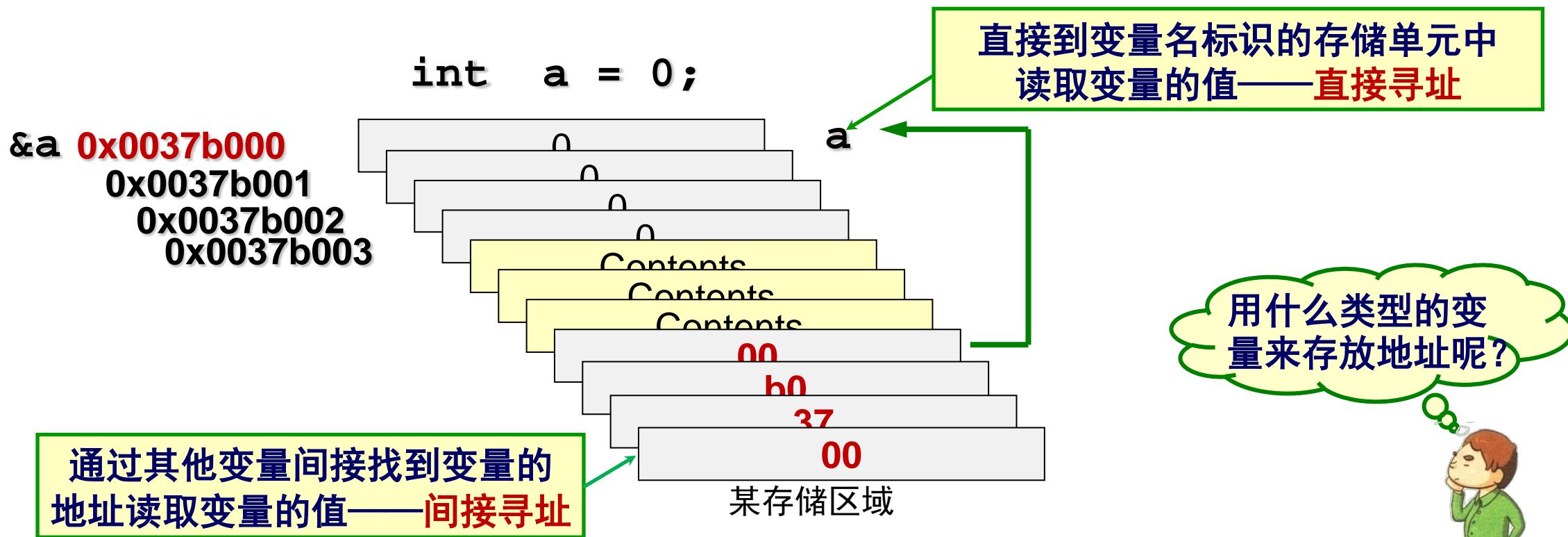
- 问题：输入数据时忘记使用取地址运算符&，这样会如何？

```
#include <stdio.h>
int main()
{
    int a = 0;
    scanf("%d", a);
    printf("a=%d\n", a);
    return 0;
}
```



a的值被当作地址。如a值为100，则输入的整数就会从地址100开始写入内存

# 如何对变量进行寻址？





# 用什么类型的变量来存放变量的地址？

- 指针（Pointer）类型
- 指针变量——具有指针类型的变量
  - \* 保存32位地址值——sizeof(pa)是4个字节
  - \* 用什么数据类型去理解它所指向的存储单元中的数据呢？

\* `int *pa;`

`int *pa = &a;`

`int a = 1; pa = &a;`

`&a 0x0028ff08`

1

0

0

0

08

ff

28

00

a

pa

指针变量指向的数据类型, 称为基类型

指针变量的内容: `&a`  
→ 指针变量pa是变量a的指针

某存储区域

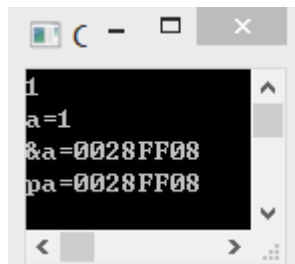


# 如何显示变量的地址？

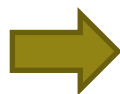
## ■ 使用%p格式符

```
#include <stdio.h>
int main()
{
    int a = 0;
    int *pa;

    printf("a=%d\n", a);
    printf("&a=%p\n", &a);
    printf("pa=%p\n", pa);
    return 0;
}
```



1  
a=1  
&a=0028FF08  
pa=0028FF08



Watches		
Function ar		
Locals		
a	1	
pa	0x28ff0c	

指针变量只能指向**同一基类型**的变量

```
#include <stdio.h>
int main()
{
    int a = 0;
    int *pa = &a;

    printf("a=%d\n", a);
    printf("&a=%p\n", &a);
    printf("pa=%p\n", pa);
    return 0;
}
```

# 使用未初始化的指针会怎样？

- 指针变量使用之前必须初始化
- 若你不知把它指向哪里，那就指向**NULL**（在stdio.h中定义为0）

```
#include <stdio.h>
int main()
{
    int a = 0;
    int *pa;

    printf("a=%d\n", a);
    printf("&a=%p\n", &a);
    printf("pa=%p\n", pa);
    return 0;
}
```

warning: 'pa' is used uninitialized



```
#include <stdio.h>
int main()
{
    int a = 0;
    int *pa = NULL;
    .....
    return 0;
}
```



# NULL是什么？

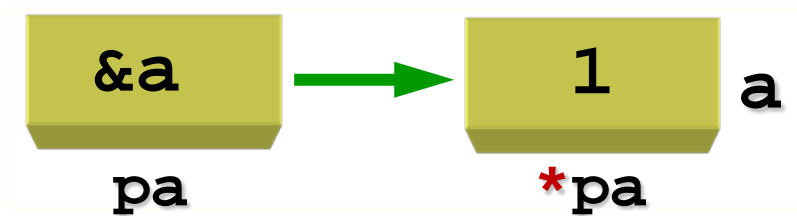
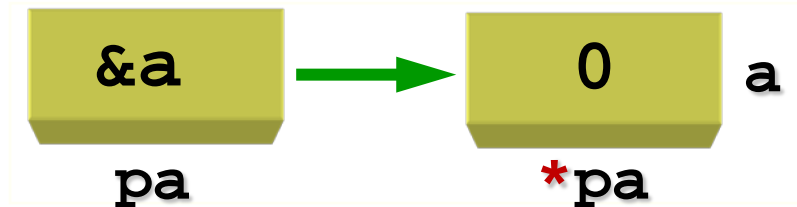
- **空指针**——值为**NULL**的指针，即无效指针
- **问题：**  $p = 0$  和  $p = \text{NULL}$  有什么区别吗？
  - \*  $p = \text{NULL}$ 可以明确地说明 $p$ 是指针变量，而不是一个数值型变量
- **问题：** 空指针就是指向地址为0的存储单元的指针吗？
- **答案：** 不一定
  - \* **并非所有编译器都使用0地址**
  - \* **某些编译器为空指针使用不存在的内存地址**



# 如何访问指针变量指向的存储单元中的数据？

- 通过**间接寻址运算符**访问（引用）指针变量指向的变量的值
- ——**指针的解引用(Pointer Dereference)**

```
#include <stdio.h>
int main()
{
    int a = 0;
    int *pa = &a;
    *pa = 1;
    printf("a=%d\n", a);
    printf("*p=%d\n", *pa);
    return 0;
}
```



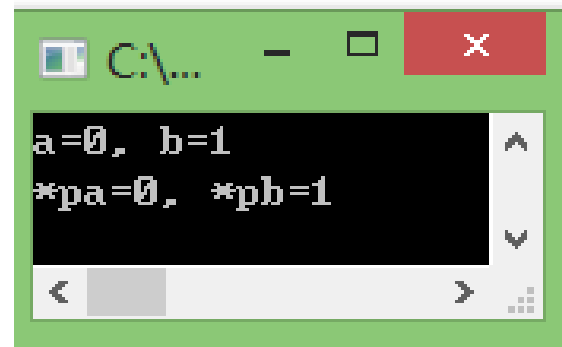
为什么要用指针？



# 指针变量的定义和初始化

```
#include <stdio.h>
int main()
{
    int  a = 0, b = 1;
    int  *pa, *pb;
    pa = &a;
    pb = &b;
    printf("a=%d, b=%d\n", a, b);
    printf("*pa=%d, *pb=%d\n", *pa, *pb);
    return 0;
}
```

不能写成: `int* pa, pb;`



The screenshot shows a Windows command prompt window with a green title bar. The window title is "C:\...". The command prompt displays the output of the C program: "a=0, b=1" and "\*pa=0, \*pb=1". The window has standard Windows window controls (minimize, maximize, close) in the top right corner.

## 讨论

- **国际C语言混乱代码大赛（IOCCC）**要求参赛者写出最有创意的最让人难以理解的C语言代码，并限制在4 kilobytes以内。第20届IOCCC获得最佳展示奖的是 Don Yang的作品（详细说明地址 <http://www.ioccc.org/2011/akari/hint.html>）。对于这样的大赛及其获奖作品，你怎么看？

[illegible]