第十章-指针

指针的概念

地址的概念

在内存区域每一个编号的地址就是“地址”。而内存单元存储的信息就是单元内容。

举例说明：

```
int a = 3;
char ch = 'a';
float f = 2.0;
```

在以上的语句中，假设这三个变量是从存储单元序号为2000 的存储单元开始的，那么编译时系统就会分配2000~2003这4个存储单元给a（因为它是int类型的，占四个字节）；分配2004号存储单元给ch（因为他是char类型的，只占一个字符）；分配2005~2008给f（因为float类型的变量占用四个字节长度）。

int类型变量的存储：比如上述的a = 3，在内存中实际存储是：2000~2003四个字节分别存储的数值是二进制：0011。在程序对该变量进行读取的时候，程序会依次读取2000~2003存储单元，然后转换正十进制数使用。

指针

概念：一个变量的地址就是指针。举例：i变量的存储地址是2000 ，那么2000这个地址就是变量i的指针。

指针变量：专门用来存放另一个变量的指针，则称它为指针变量。

变量的指针和指向变量的指针变量

> &：读作“取地址运算符”，含义是，取变量的地址。int i = 3;　　那么 &i就是取变量i的地址。
> *：读作“指向”，取指针所指向的内存单元的内容。i_p是指向i的指针变量，那么*i_p就是取i_p所指向的指针的内容。

这部分内容比较绕，实际上&和*就是两个互逆运算罢了。以下两个表达式是等价的：

```
i = 3;
*i_p = 3;
```

因为两者指向的实际都是i这个变量的内容跟。

定义一个指针变量

C语言中规定，所有的变量在使用之前都需要进行定义。不像一些基本类型，指针变量是用来存储地址的，因此需要将其定义为指针类型的变量。一般格式为：

数据类型 *指针变量名; //指针变量名一般使用与point有关的标识符，such as pointer_1、p1、i_p等等。

举例说明：

```
int i, j;
int *pointer_1, pointer_2;
```

另外可以用赋值语句是一个指针变量得到另一个变量的地址，从而使它指向该变量。

```
pointer_1 = &i;
pointer_2 = &j;
```

说明：

（1） 在定义指针变量时前面的“*”表示，当前定义的变量是指针类型。类比数组类型的定义：

```
int arr[] = {1, 2, 3};  数组的定义中 [] 就可以表示这是个数组，这是一种标识
int *i_p = &i;     指针定义中 * 与数组中的 [] 类似，也是作为一种标识
```

（2） 在定义指针变量的时候必须指明变量类型，这是必须的。

指针变量的引用

指针变量的引用需要指针变量运算符。

举例说明：

```
#include "stdio.h"
void main(){
```

```
    int a, b;
    int *pointer_1, *pointer_2;
    pointer_1 = &a;
    pointer_2 = &b;

    printf("Please input a,b:");
    scanf("%d%d", &a, &b);
    printf("a = %d, b = %d", a, b);

    printf("Again input a, b:");
    scanf("%d%d", pointer_1, pointer_2);
    printf("*pointer_1 = %d, *pointer_2 = %d\n",  *pointer_1, *pointer_2);
}
```

指针变量作为函数参数

直接上例子:

```
#include "stdio.h"
int main(void){
    void swap(int *p1, int *p2);
    int a, b;
    int *pointer_1, *pointer_2;

    scanf("%d,%d", &a, &b);
    pointer_1 = &a;
    pointer_2 = &b;
    if (a > b){
        swap(pointer_1,pointer_2);  // 调用swap函数的时候传递的实参是指针
    }
    printf("\n%d,%d\n", a, b);

    return 0;
}

void swap(int *p1, int *p2){
    int temp;
    temp = *p1;
    *p1 = *p2;  //在函数中，通过指针间接的访问了a, b
    *p2 = temp;
}
```

数组与指针

指向数组元素的指针

```
首先来定义一个数组
int a[10];  //定义a为包含10个整型数据的数组
int *p;  //定义一个指针变量
p = &a[0];  //将下标为0的元素的地址赋给指针变量p
```

注：在C语言中规定，数组名表示数组的首地址。是地址常量。数组中的首元素的地址也是数组的首地址。因此一下两个语句意义想同：

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

需要注意的是：这里的a并不代表整个数组，而是代表这个数组的首地址，p是一个指针变量。

通过指针引用数组元素

```
int *p;
int a[10];
p = &a[0];
*p = 1;
```

以上程序段中，最后一句就是将1赋值给指针变量p所指向的元素。程序段中出现了两次*p，这两次的出现有着本质的区别，第一次是定义一个指针变量，第二次是为了解析p所代表的地址。

说明：

在C语言中规定：如果指针变量p指向某一个元素，则p+1指向的是同一数组的下一个元素，并不是对指针变量p所代表的地址进行数学运算。

字符串与指针

字符串的表达形式

在C语言中可以有两种方法访问一个字符串

（1） 用字符数组存放字符串，然后进行输出。

（2） 用字符指针指向字符串。

```
#include "stdio.h"
void main(void){
    char *string = "I love China!";
    printf("%s\n", string);
}
```

一定要时刻清楚，在C语言中是没有字符串的概念的，所以上述语句中，char *string = " I love China! "; 的逻辑是，先创建一个字符数组用来存放具体元素，然后将字符数组的首地址赋值给string这个指针变量。

指向函数的指针

引言：一个函数在编译时被指定一个入口地址。这个函数入口地址就称为函数的指针。

```
# include "stdio.h"
int main(void){
    int max(int, int);
    int a, b, c;
    scanf("%d, %d", &a, &b);
    c = max(a, b);
    printf("a = %d, b = %d, max = %d", a, b, c);
}
int max(int x, int y){
    int z;
    if (x > y){
        z = x;
    }else{
        z = y;
    }
    return z;
}
```

为了节省内存单元，将main函数改写为如下：

```
int main(void){
    int max (int,int );
    int (*p)(int, int);    // 在这里定义了一个指向函数的指针变量
    int a, b, c;
    p = max;    // 在这里将函数max的初始地址复制给指针变量p
    scanf("%d, %d", &a, &b);
    c = (*p)(a, b);
    printf("a = %d, b = %d, max = %d", a, b, c);
}
```

用指向函数的指针作为函数参数

这个实际上就是将一个函数的运行结果作为参数传递给另一个函数使其正常运行。

返回指针值的函数