理解函数指针和typedef函数指针

tracy_668 关注
2020.05.14 23:51:42 字数 1,165 阅读 610

函数指针

```
void myFun(int x);   //声明也可写成 void myFun( int );
int main()
{
    myFun(100);//一般的函数调用
    return 0;
}
void myFun(int x)
{
    printf("myFun: %d\n",x);
}
```

我们一开始只是从功能上或者说从数学意义上理解myFun这个函数，知道myFun函数名代表的是一个功能（或是说一段代码），函数名到底又是什么东西呢？

函数指针变量：

一个数据变量的内存地址可以存储在相应的指针变量中，函数的首地址也以存储在某个函数指针变量中。这样，我就可以通过这个函数指针变量来调用指向的函数了。

在C系列语言中，任何一个变量，总是要先声明，之后才能使用的，函数指针变量也应该要先声明。

函数指针变量的声明：

void (funP)(int) ;*//声明一个指向同样参数、返回值的函数指针变量。*

（整个函数指针变量的声明格式如同函数myFun的声明处一样，只不过——我们把myFun改成（funP）而已。这样就有了一个能指向myFun函数的指针了。当然，这个funP指针变量也可以指向所有其他同参数及返回值的函数。）

```
#include <stdio.h>
#include <stdlib.h>

void (*funP)(int);    //声明也可写成void(*funP)(int x), 但习惯上一般不这样。
void (*funA)(int);
void myFun( int x);   //声明也可写成 void myFun( int );
int main()
{
    //一般的函数调用
    myFun(100);

    //myFun与funP的类型关系类似于int 与int *的关系。
    funP=&myFun;  //将myFun函数的地址赋给funP变量
    (*funP)(200);  //通过函数指针变量来调用函数

    //myFun与funA的类型关系类似于int 与int 的关系。
    funA=myFun;
    funA(300);

    //三个输出语句的输出的调用
    funP(400);
    (*funA)(500);
    (*myFun)(1000);

    return 0;
}

void myFun(int x)
{
    printf("myFun: %d\n",x);
}
```

总结：

1、 其实，myFun的函数名与funP、funA函数指针都是一样的，即都是函数指针。myFun函数名是一个函数指针常量，而funP、funA是函数指针变量，这是它们的关系。

2、函数名调用如果都得如{myFun(10)这样，那书写与读起来都是不方便和不习惯的。所以C语言的设计者们才会设计成又可在funP(10)这种形式地调用（这样方便多了，并与数学中的函数形式一样）。

3、为了统一调用方式，funP函数指针变量也可以funP(10)的形式来调用。

4、就是说，可以写成funP=&myFun形式，也可以写成funP=myFun。

5、但是在声明时，void myFun(int)不能写成void (myFun)(int)。void (funP)(int)不能写成void funP(int)。

6、函数指针变量也可以存入一个数组内。数组的声明方法：int (fArray[10]) (int);

```
#include <stdio.h>
#include <stdlib.h>

void (*funP)(int);
void (*funA)(int);
void myFun( int x);
int main()
{
    funP=&myFun;
    //深入理解
    printf("sizeof(myFun)=%d\n",sizeof(myFun));
    printf("sizeof(funP)=%d\n",sizeof(funP));
    printf("myFun= 0x%p=0x%p\n",&myFun,myFun);
    printf("funP= 0x%p=0x%p\n",&funP,funP);
    printf("funA= 0x%p=0x%p\n",&funA,funA);
    return 0;
}

void myFun(int x)
{
    printf("myFun: %d\n",x);
}
```

总结：

1、函数指针变量跟普通的指针一样在32位系统下大小都为4，但是函数指针常量的大小为1.

2、函数指针变量和函数指针常量存储在内存的不同位置。

3、为负值的函数指针变量（全局）的值为0。

函数指针作为某个函数的参数

既然函数指针变量是一个变量，当然也可以作为某个函数的参数来使用的。

示例：

```
#include <stdio.h>
#include <stdlib.h>

typedef void(*FunType)(int);
//可以一个typedef声明符，这样就定义一个名为FunType函数指针类型，而不是一个FunType变量。
//形式同 typedef int* PINT;
void myFun(int x);
void hisFun(int x);
void herFun(int x);
void callFun(FunType fp,int x);
int main()
{
    callFun(myFun,100);//传入函数指针常量，作为回调函数
    callFun(hisFun,200);
    callFun(herFun,300);

    return 0;
}

void callFun(FunType fp,int x)
{
    fp(x);//通过fp的指针执行传递进来的函数，注意fp所指的函数有一个参数
}

void myFun(int x)
{
    printf("myFun: %d\n",x);
}
void hisFun(int x)
{
    printf("hisFun: %d\n",x);
}
void herFun(int x)
{
    printf("herFun: %d\n",x);
}
```

1.简单的函数指针的应用

形式1：返回类型(*函数名)(参数表)

```
char (*pFun)(int);
char gFun(int a){ return;}
void main()
{
    pFun = gFun;
    (*pFun)(2);
}
```

第一行定义了一个指针变量pFun，首先我们根据前面提到的"形式1"认识到它是一个指向某种函数的指针，这种函数参数是一个int型，返回值是char类型。只有第一句我们还无法使用这个指针，因为我们还未对它进行赋值。

第二行定义了一个函数gFun()，该函数至少是一个以int为参数返回char的函数，我们要从指针的层次上理解函数——函数的函数名实际上就是一个指针，函数名指向该函数的代码在内存中的首地址

然后在main()函数了，它的第一句也应该看得懂了——它将函数gFun的地址赋值给变量pFun。main()函数的第二句中**pFun*是然要取pFun所指向地址的内容，当然也就是取出了函数gFun()的内容，然后给定参数为2。

2.使用typedef更直观更方便

形式1：typedef 返回类型(*新类型)(参数表)

```
typedef char (*PTRFUN)(int);
PTRFUN pFun;
char gFun(int a){ return;}
void main()
{
    pFun = gFun;
    (*pFun)(2);
}
```

typedef的功能是定义新的类型。第一句就是定义了一种PTRFUN的类型，并定义这种类型为指向某种函数的指针，这种函数以一个int为参数并返回char类型。后面就可以像使用int,char一样使用PTRFUN了。

第二行的代码便使用这个新类型定义了变量pFun，此时就可以像使用形式1一样使用这个变量了。

```
#include <stdio.h>
#include <assert.h>

typedef int (*FP_CALC)(int,int);//定义一个函数指针类型

int add(int a, int b)
{
    return a + b;
}

int sub(int a, int b)
{
    return a - b;
}

int mul(int a, int b)
{
    return a * b;
}

int div(int a, int b)
{
    return b ? a/b : -1;
}
//定义一个函数，参数为op，返回一个指针，这指针类型为调用有两个int参数，
//返回类型为int的函数指针，它的作用是根据操作符返回相应函数的地址
FP_CALC calc_func(char op)
{
    switch( op )
    {
    case '+':
        return add;
    case '-':
        return sub;
    case '*':
        return mul;
    case '/':
        return div;
    default:
        return NULL;
    }
    return NULL;
}
//s_calc_func为函数，它的参数是 op,
//返回值为一个函数指针(int参数，返回类型为int的函数指针
int (*s_calc_func(char op)) (int , int)
{
    return calc_func(op);
}

//最终用户直接调用的函数，该函数接收两个int型数，
//另一个是术运算符，返回两数的运算结果
int calc(int a, int b, char op)
{
    FP_CALC fp = calc_func(op);
    int (*s_fp)(int,int) = s_calc_func(op);//用于测试

    assert(fp == s_fp);// 可以断言证两个是相等的

    if(fp)
        return fp(a,b);
    else
        return -1;
}
void main()
{
    int a = 100, b = 20;

    printf("calc(%d, %d, %c) = %d\n", a, b, '+', calc(a, b, '+'));
    printf("calc(%d, %d, %c) = %d\n", a, b, '-', calc(a, b, '-'));
    printf("calc(%d, %d, %c) = %d\n", a, b, '*', calc(a, b, '*'));
    printf("calc(%d, %d, %c) = %d\n", a, b, '/', calc(a, b, '/'));
}
```


tracy_668
总资产61 共写了197.3W字 获得2,285个赞 共853个粉丝
关注

热门故事
当了三年上门女婿备受屈辱，今天我终于做了一回男人
一个故事告诉你，古代后宫不受宠的女人有多惨
如果最后那个人不是你，那么是谁都可以
当女人出现这些行为，说明她想放弃婚姻了
查看更多

推荐阅读
C++<第二十七篇> 函数模板 阅读 243
C++<第九篇> 指针 阅读 125
C++<第三十二篇> 转换构造函数和类型转换函数 阅读 154
golang—4：变量声明和初始化 阅读 131
Swift基础语法（三）函数 阅读 168