前言：

typeof关键字是C语言中的一个新扩展，这个特性在linux内核中应用非常广泛。

只在linux 下有效，作用是 复制数据类型，例如：typeof（int）i; 就等于 int i;

一，说明

typeof的参数可以是两种形式：表达式或类型。

1，表达式的的例子：

```
typeof(x[0](1)
```

这里假设x是一个函数指针数组，这样就可以得到这个函数返回值的类型了。

如果将typeof用于表达式，则该表达式不会执行。只会得到该表达式的类型。

以下示例声明了int类型的var变量，因为表达式foo()是int类型的。由于表达式不会被执行，所以不会调用foo函数。

```
extern int foo();
typeof(foo()) var;
```

2，参数的例子：

```
typeof(int *) a,b;
```

等价于：

```
int *a,*b;
```

二，实例

1，把y定义成x指向的数据类型：

```
typeof(*x) y;
```

2，把y定义成x指向数据类型的数组：

```
typeof(*x) y[4];
```

3，把y定义成一个字符指针数组：

```
typeof(typeof(char *)[4] y;
```

这与下面的定义等价：

```
char *y[4];
```

```
4, typeof(int *) p1,p2; /* Declares two int pointers p1, p2 */
   int *p1, *p2;

5, typeof(int) *p3,p4;/* Declares int pointer p3 and int p4 */
   int *p3, p4;

6, typeof(int [10]) a1, a2;/* Declares two arrays of integers */
   int a1[10], a2[10];
```

三，局限

typeof构造中的类型名**不能**包含存储类说明符，如extern或static。不过允许包含类型限定符，如const或volatile。

例如，下列代码是无效的，因为它在typeof构造中声明了extern：

```
typeof(extern int) a;
```