由于目录本身也是文件，因此它们也有大小。设我们想要计算被该树所有文件占用的磁盘区块的总数。最自然的做法是找出含于子目录/usr/mark(30)、/usr/alex(9)和/usr/bill(32)的区块的个数。于是，磁盘区块的总数就是子目录中的区块的总数(71)加上/usr使用的一个区块，共72个区块。图4-9中的伪码方法size实现这种遍历策略。

```
    public int size( )
    {
1       int totalSize = sizeOfThisFile( );

2       if( isDirectory( ) )
3           for each file c in this directory (for each child)
4               totalSize += c.size( );

5       return totalSize;
    }
```

图4-9　计算一个目录大小的伪码例程

如果当前对象不是目录，那么size只返回它所占用的区块数。否则，被该目录占用的区块数将被加到在其所有子节点(递归地)发现的区块数中去。为了区别后序遍历策略和先序遍历策略之间的不同，图4-10显示每个目录或文件的大小是如何由该算法产生的。

```
            ch1.r            3
            ch2.r            2
            ch3.r            4
        book                10
                    syl.r    1
                fall05       2
                    syl.r    5
                spr06        6
                    syl.r    2
                sum06        3
            cop3530         12
        course              13
        junk                 6
    mark                    30
        junk                 8
    alex                     9
        work                 1
                    grades   3
                    prog1.r  4
                    prog2.r  1
                fall05       9
                    prog2.r  2
                    prog1.r  7
                    grades   9
                fall06      19
            cop3212         29
        course              30
    bill                    32
/usr                        72
```

图4-10　函数size的印迹