首先你要明白typedef的作用。typedef是C语言里的一个关键字，用来给一个类型别名（化名）。比如

|  |  |
| --- | --- |
| 1  2  3 | **typedef** **int** [size\_t](https://www.baidu.com/s?wd=size_t&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao);    **typedef** **struct** node Node; |

第一个例子给内置类型int一个别名[size\_t](https://www.baidu.com/s?wd=size_t&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao)。第二个例子给struct node一个别名Node。这里你看到了typedef的两个常用情况。第一个例子用以给一个通用类型更具化的别名，这往往会增加你的代码的可读性。第二个例子用以简化代码的繁复性，每当你需要声明一个struct node变量的时候，你可以直接用Node。

你的例子

|  |  |
| --- | --- |
| 1  2  3  4 | **typedef** **struct** node {  **struct** library data;  **struct** node \*next;  } Node, \* Link; |

只是一个比较紧凑的写法，等价于如下写法：

|  |  |
| --- | --- |
| 1  2  3  4  5 | **typedef** **struct** node Node, \* Link;  **struct** node {  **struct** library data;    Link next;  // Node \* next;  } |

注意我把typedef置于结构体node的定义之前（这是一个C成语）。它的好处是C编译器会先看见别名定义，所以在结构体node的定义里面我就已经可以用其别名Link或者Node来声明变量或指针了。你可以把typedef置于结构体node的定义之后，写作

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1  2  3  4  5 | **struct** node {  **struct** library data;  **struct** node \* next;  }  **typedef** **struct** node Node, \* Link; |  |  |  |

但这样你就不能在node的定义里面用定义完才给的别名Node或Link了。所以你看到里面next的声明用的是struct node \*。

最后，

|  |  |
| --- | --- |
| 1 | **typedef** **struct** node Node, \* Link; |

仍然是一个紧凑的写法，等价于：

|  |  |
| --- | --- |
| 1  2 | **typedef** **struct** node Node;  **typedef** **struct** node \* Link; |

第一条语句给struct node（结构体node本身）一个别名Node。第二条语句给struct node \*（指向node结构体的指针）一个别名Link。