```
    long fun(struct ELE *ptr)
    ptr in %rdi
1   fun:
2     movl    $0, %eax            result = 0
3     jmp     .L2                 Goto middle
4   .L3:                        loop:
5     addq    (%rdi), %rax        result += ptr->v
6     movq    8(%rdi), %rdi       ptr = ptr->p
7   .L2:                        middle:
8     testq   %rdi, %rdi          Test ptr
9     jne     .L3                 If != NULL, goto loop
10    rep; ret
```

A. 根据加了注释的代码，可以得到C语言：

```
long fun(struct ELE *ptr) {
    long val = 0;
    while (ptr) {
        val += ptr->v;
        ptr  = ptr->p;
    }
    return val;
}
```

B. 可以看到每个结构都是一个单链表中的元素，字段v是元素的值，字段p是指向下一个元素的指针。函数fun计算列表中元素值的和。

3.43　结构和联合涉及的概念很简单，但是需要练习来习惯不同的引用模式和它们的实现。

表达式	类型	代码
up->t1.u	long	movq(%rdi),%rax movq %rax,(%rsi)
up->t1.v	short	movw 8(%rdi),%ax movw %ax,(%rsi)
&up->t1.w	char*	addq $,%rdi movq %rdi,(%rsi)
up->t2.a	int*	movq %rdi,%rsi

up->t2.a[up- > t1.u]	int	movq (%rdi),%rax
		movl (%rdi,%rax,4),%eax
		movl %eax,(%rsi)

*up->t2.p	char	movq 8(%rdi),%rax
		movb (%rax),%al
		movb %al,(%rsi)

3.44　想理解各种数据结构需要多少存储，以及编译器为访问这些结构产生的代码，理解结构的布局和对齐是非常重要的。这个练习让你看清楚一些示例结构的细节。

A. struct P1 { int i; char c; int j; char d; };

i	c	j	d	总共	对齐
0	4	8	12	16	4

B. struct P2 { int i; char c; char d; long j; };

i	c	d	j	总共	对齐
0	4	5	8	16	8