2.6 已知 L 是无表头结点的单链表，且 P 结点既不是首元结点，也不是尾元结点，试从下列提供的答案中选择合适的语句序列。

a. 在 P 结点后插入 S 结点的语句序列是_____。

b. 在 P 结点前插入 S 结点的语句序列是_____。

c. 在表首插入 S 结点的语句序列是_____。

d. 在表尾插入 S 结点的语句序列是_____。

(1) P->next=S;
(2) P->next=P->next->next;
(3) P->next=S->next;
(4) S->next=P->next;
(5) S->next=L;
(6) S->next=NULL;
(7) Q=P;
(8) while(P->next!=Q) P=P->next;
(9) while(P->next!=NULL) P=P->next;
(10) P=Q;
(11) P=L;
(12) L=S;
(13) L=P;

解：a. (4) (1)

b. (7) (11) (8) (4) (1)

c. (5) (12)

d. (9) (1) (6)

2.7 已知 L 是带表头结点的非空单链表，且 P 结点既不是首元结点，也不是尾元结点，试从下列提供的答案中选择合适的语句序列。

a. 删除 P 结点的直接后继结点的语句序列是_____。

b. 删除 P 结点的直接前驱结点的语句序列是_____。

c. 删除 P 结点的语句序列是_____。

d. 删除首元结点的语句序列是_____。

e. 删除尾元结点的语句序列是_____。

(1) P=P->next;
(2) P->next=P;
(3) P->next=P->next->next;
(4) P=P->next->next;
(5) while(P!=NULL) P=P->next;
(6) while(Q->next!=NULL) { P=Q; Q=Q->next; }
(7) while(P->next!=Q) P=P->next;
(8) while(P->next->next!=Q) P=P->next;
(9) while(P->next->next!=NULL) P=P->next;
(10) Q=P;
(11) Q=P->next;
(12) P=L;
(13) L=L->next;
(14) free(Q);

解：a. (11) (3) (14)
b. (10) (12) (8) (3) (14)
c. (10) (12) (7) (3) (14)
d. (12) (11) (3) (14)
e. (9) (11) (3) (14)

2.8 已知 P 结点是某双向链表的中间结点，试从下列提供的答案中选择合适的语句序列。

a. 在 P 结点后插入 S 结点的语句序列是_____。
b. 在 P 结点前插入 S 结点的语句序列是_____。
c. 删除 P 结点的直接后继结点的语句序列是_____。
d. 删除 P 结点的直接前驱结点的语句序列是_____。
e. 删除 P 结点的语句序列是_____。

(1) P->next=P->next->next;
(2) P->priou=P->priou->priou;
(3) P->next=S;
(4) P->priou=S;
(5) S->next=P;
(6) S->priou=P;
(7) S->next=P->next;
(8) S->priou=P->priou;
(9) P->priou->next=P->next;
(10) P->priou->next=P;
(11) P->next->priou=P;
(12) P->next->priou=S;
(13) P->priou->next=S;
(14) P->next->priou=P->priou;
(15) Q=P->next;
(16) Q=P->priou;
(17) free(P);
(18) free(Q);

解：a. (7) (3) (6) (12)
b. (8) (4) (5) (13)
c. (15) (1) (11) (18)
d. (16) (2) (10) (18)
e. (14) (9) (17)

2.9 简述以下算法的功能。

```
(1) Status A(LinkedList L) { //L 是无表头结点的单链表
        if(L && L->next) {
            Q=L;       L=L->next;P=L;
            while(P->next) P=P->next;
            P->next=Q;Q->next=NULL;
        }
        return OK;
    }
```

(2)
```
void BB(LNode *s, LNode *q) {
        p=s;
        while(p->next!=q) p=p->next;
        p->next =s;
    }
    void AA(LNode *pa, LNode *pb) {
        //pa 和 pb 分别指向单循环链表中的两个结点
        BB(pa,pb);
        BB(pb,pa);
    }
```

解：(1) 如果 L 的长度不小于 2，将 L 的首元结点变成尾元结点。

(2) 将单循环链表拆成两个单循环链表。