四、基础知识题

2.1① 描述以下三个概念的区别：头指针，头结点，首元结点(第一个元素结点)。

2.2① 填空题。

(1) 在顺序表① 中插入或删除一个元素，需要平均移动_____元素，具体移动的元素个数与_____有关。

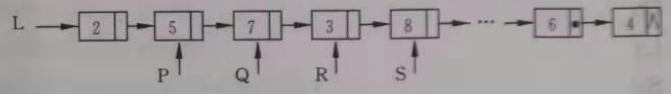
(2) 顺序表中逻辑上相邻的元素的物理位置_____紧邻。单链表中逻辑上相邻的元素的物理位置_____紧邻。

(3) 在单链表中，除了首元结点外，任一结点的存储位置由_____指示。

(4) 在单链表中设置头结点的作用是_____。

2.3② 在什么情况下用顺序表比链表好？

2.4① 对以下单链表分别执行下列各程序段，并画出结果示意图。



(1) Q=P->next;

(2) L=P->next;

(3) R->data=P->data;

(4) R->data=P->next->data;

(5) P->next->next->next->data=P->data;

(6) T=P;

```
while (T!=NULL) { T->data=T->data * 2;  T=T->next;
```

(7) T=P;

```
while (T->next!=NULL) { T->data=T->data * 2;  T=T->next;
```

2.5① 画出执行下列各行语句后各指针及链表的示意图。

```
L=(LinkList)malloc(sizeof(LNode));  P=L;
for (i=1; i<=4; i++) {
    P->next = (LinkList)malloc(sizeof(LNode));
    P = P->next;  P->data = i * 2-1;
}
P->next=NULL;
for (i=4; i>=1; i--;) Ins_LinkList(L, i+1, i * 2);
```

① 在本书中，顺序表即为采用顺序存储结构的线性表。

(i=1; i<=3 ; i++) Del_LinkList(L

②　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

下列提供的答案中选择合适的语句序列。

在 P 结点后插入 S 结点的语句序列是 (4)(1)。

在 P 结点前插入 S 结点的语句序列是 (7)(11)(8)(4)(1)

在表首插入 S 结点的语句序列是 (5)(12)。

在表尾插入 S 结点的语句序列是 (11)(9)(1)(6)

(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;

2.7②　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，

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 (11)(3)(14)。

b. 删除 P 结点的直接前驱结点的语句序列是 (10)(12)(8)(3)(14)

c. 删除 P 结点的语句序列是 (10)(12)(7)(3)(14)

d. 删除首元结点的语句序列是 (12)(13)(10)(14)

e. 删除尾元结点的语句序列是

(1) P=P->next;

(2) P->next=P

(12) P = L ;

(13) L= L->next ;

(14) free(Q);

2.8② 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语句序列。

a. 在P结点后插入S结点的语句序列是 (7)、(12)、(3)、(6)。

b. 在P结点前插入S结点的语句序列是 (5)、(8)、(13)、(15)。

c. 删除P结点的直接后继结点的语句序列是 (1)、(11)、(18)。

d. 删除P结点的直接前驱结点的语句序列是 (16)、(2)、(10)、(18)。

e. 删除P结点的语句序列是 (3)、(8)、(13)、(17)。

(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) ;

```cpp
//2.11
#include<bits/stdc++.h>
using namespace std;

int va[101],n,x,pos;

void insert(int val){
    int idx=0;
    for(;idx<n;idx++) if(va[idx]>=val) break;
    for(int i=n;i>idx;i--) va[i]=va[i-1];
    va[idx]=val;
}

signed main(){
    cin>>n;//顺序表元素个数
    for(int i=0;i<n;i++) cin>>va[i]; //输入顺序表中的元素
    sort(va,va+n); //保证递增有序
    cin>>x;
    insert(x); //将x插适当的位置
    for(int i=0;i<=n;i++) cout<<va[i]; //输出顺序表
    return 0;
}
```

```cpp
//2.15
#include<bits/stdc++.h>
using namespace std;

struct node{
    int data;
    node *nxt;
};

struct List{
    int length;
    node *head,*tail;
}A,B;//定义两个链表A,B

signed main(){
    node ha=A->head; //定义两者的头节点
    node hb=B->head;
    A.length=m; //两链表的长度
    B.length=n;
    A->tail->nxt=B->head;//将B接在A的尾部
    A.length+=B.length();//长度增加
    node hc=A->head; //新的头节点
    return 0;
}

```