一、单选题

1．下列程序段的时间复杂度是（　　）？

```
count=0;
for(j=1;j<=n;j++)
    for(k=1;k<=n;k*=3)
        count++;
```

A．O($\log_3 n$)

B．O(n)

C．O($n\log_3 n$)

D．O($n^2$)

解：设内层循环执行 m 次，则有$3^m \le n$，因此$m \le \log_3 n$；外层循环执行 n 次，所以两层循环执行次数$\le n\log_3 n$，大 O 记法写为$O(n\log_3 n)$

2．下面代码的功能是（　）？

```
3   double E01(double *a, int n, double x)
4   {//a={a0,a1,a2,...,an}; n为阶数，最高次幂;
5       double sum = a[0], item = x;//初始化为常数项
6
7       for (int i = 1; i <= n; i++)
8       {
9           sum += a[i] * item;
10          item *= x;
11      }
12      return sum;
13  }
```

A．计算$P_n(x) = \sum_{i=0}^{n} a_i x$

B．计算$P_n(x) = \sum_{i=0}^{n} a_i$

C．计算$P_n(x) = \sum_{i=0}^{n} x^i$

D．计算$P_n(x) = \sum_{i=0}^{n} a_i x^i$

解：初始，sum=a[0]，即$a_0x^0$，item=x；循环 n 次，第 i 次，sum 累加 a[i]*item=a[i]*$x^i$，item 更新为 item*x=$x^{i+1}$，从而为下一次累加做准备。所以，第 i 次 sum=sum+$a_ix^i$。因此，代码段功能为计算多项式，应选 D。

3．已知 L 是带表头结点的非空链表，且 P 结点既不是首元素结点，也不是尾元素结点，语句序列中，能删除 P 结点的直接后继结点的语句序列是(　　)。

(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);

A．(10)(12)(8)(11)(3)(14)
B．(12) (11)(3)(14)
C．(11)(3)(14)
D．(9)(11)(3)(14)

解：结点 P 的直接后继为 P->next，仅从链表的关系而言，把 P 的直接后继 P->next 移除链表，只要更新结点 P 的后继指针 P->next，使之指向当前后继的后继即可，P->next = P->next->next。需要注意的是，被移除的结点 P->next 的存储空间应被动态释放，所以需要先保存改地址 Q=P->next，再动态释放 free(Q)。所以正确选项是 C。
建议：可以画图明确结点关系再设计链表的操作。

## 二、判断题

1．“完全二叉树的某结点若无左孩子，则它必是叶结点。”说法正确吗？。（ ）
解：正确。
根据完全二叉树的特点，若无左孩子，必定没有右孩子。

## 三、填空题

1．在长度为 n 的顺序表中插入或删除一个元素，平均约需要移动_____元素，具体移动的元素个数与_____有关。
解：
在顺序表中插入或删除一个元素，平均约需要移动 表中一半 元素，具体移动的元素个数与 表长和该元素在表中的位置 有关。

## 四、主观题

1．试写一算法，对单链表实现就地逆置
解：对于线性表 a0,a1,a2,…,an 而言，为其构造链表时，若每次将新结点追加到表尾，那么从表头到表尾的结点对应线性表元素 a0,a1,a2,…,an；若每次将新结点从表头插入，则那么从表头到表尾的结点与线性表元素顺序逆序，即依次对应线性表元素 an, ,…,a2,a1,a0。就地逆置利用这个特点，将结点顺序从链表中摘出，再从表头插入。

```
typedef struct node
{
    int data;
    struct node *next;
}NODE, *PNODE, *LinkList;
void Invert(LinkList head) {//逆置链表,链表已经存在，head指向其头结点。
    NODE *p = NULL,
        *q = head->next;//head指向头结点，q指向a1

    head->next = NULL;//头结点从原链表中断开
```

```
    while (q)
    {
        p = q;//摘出结点ai
        q = q->next;
        p->next = head->next;
        head->next = p;//将ai插入表头，形成逆序
    }
}
```