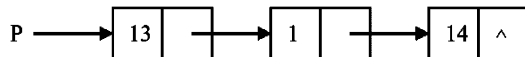
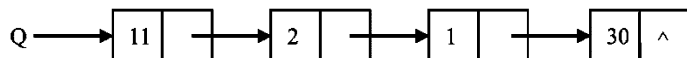


链表操作专题练习

3、用链表存储一组不重复的整数，如13、1、14三个数被存储到链表P：



编写函数SUM(P, Q, n)，参数P和Q分别指向两个链表（表示两组数），函数返回一个新链表R（注意R和P、Q的结点结构不同），表示从P、Q两组数中各取一个数，相加之和等于n的一个组合。例如当n为15，假设Q如下：



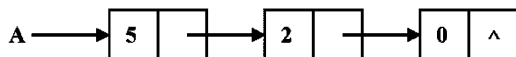
执行SUM(P, Q, 15)，结果如下（结点在链表中的排列次序任意，下图只是一例）：



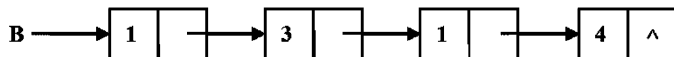
要求：（1）写出P、R链表结点定义；（2分）

（2）编写完成函数SUM，实现题目要求的功能。（12分）

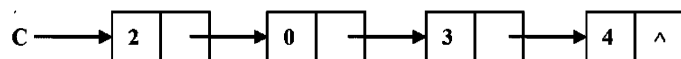
3、用链表存储一个八进制数（类型为 unsigned long）， $(520)_8$ 可存储为如下链表（^表示链表尾）：



函数Add(A, B)的参数A和B分别指向两条链表（表示两个八进制数），函数返回链表C，表示A加上B后所得的八进制数。例如，再有B如下：



则执行C=Add(A, B)后，由于 $(520)_8 + (1314)_8 = (2034)_8$ ，最后结果如下：

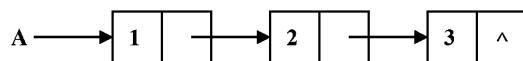


要求：（1）给出链表结点定义（2分）；

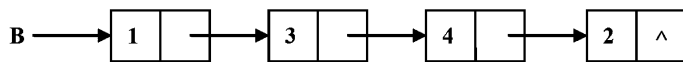
友情提示：可先转成十进制再相加：)

（2）编写Add函数，实现题目要求的功能（12分）。

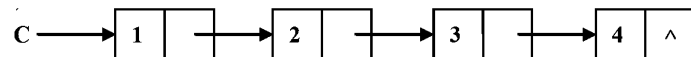
3、用链表存储一个集合。例如集合 $A=\{1, 2, 3\}$ 可存储如下 (^表示链表尾):



函数 $\text{Union}(A, B)$ 的参数 A 和 B 分别指向两条链表 (表示两个集合), 函数返回一个新链表 C (A, B 保持不变), 表示 $A \cup B$ 后所得的新集合。例如, 再有集合 B 如下:



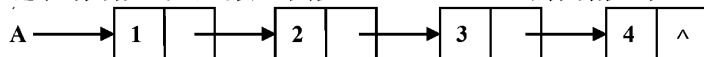
则执行 $C=\text{Union}(A, B)$ 后, 由于 $\{1, 2, 3\} \cup \{1, 3, 4, 2\} = \{1, 2, 3, 4\}$, 最后结果如下:



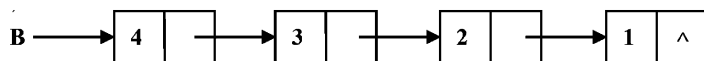
要求: (1) 给出链表结点定义 (2分);

(2) 编写 Union 函数, 实现题目要求的功能 (12分)。

3. 用链表存储一批整数。例如 1、2、3、4 可存储如下 (^表示链表尾):



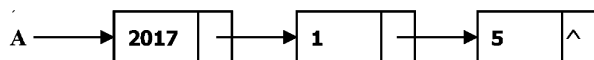
函数 $\text{Rev}(A)$ 的参数 A 指向一条链表, 函数执行后得到一个新链表 B (A 保持不变), 表示将 A 转置后所得的新链表。例如, 执行 $B=\text{Rev}(A)$ 后, 最后结果如下:



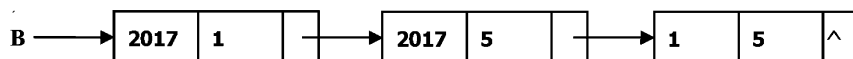
要求: (1) 给出链表结点定义 (2分);

(2) 编写 Rev 函数, 实现题目要求的功能 (12 分)。

3. 用链表存储一批各不相同的非负整数, 如 2017、1、5 可存储如下 (^表示链表尾):



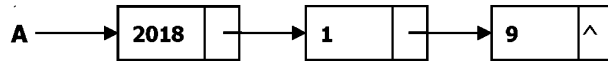
写一个函数 $\text{Pair}(A)$, 参数 A 为上述链表, 函数功能是求出 A 中所有的互质 (即最大公约数为 1) 整数对, 并将其插入到新链表 B 中, 最后返回新链表。例如, 执行 $B = \text{Pair}(A)$ 后, 最后结果如下 (结点顺序随意):



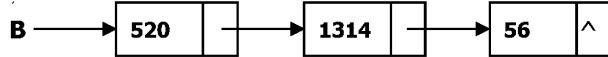
要求: (1) 给出两种链表结点 struct LA 和 struct LB 的定义 (2分);

(2) 编写 Pair 函数, 实现题目要求的功能 (12 分)。

3. 链表 A 存储了一批非负整数，如下所示 (^表示链表尾):



链表 B 也存储了一批非负整数，如下所示 (^表示链表尾):



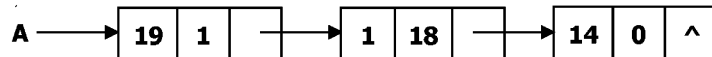
编写一个名为Pair的函数，形参A, B为上述链表，函数返回一个新链表C，C中包含了所有满足下述条件的组合：(a)从A, B中各取一个数，(b)两个数的各位数字之和相等。上例中，执行C=Pair(A, B)后结果如下（结点顺序随意）：



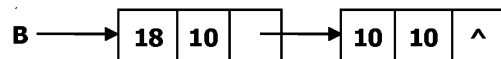
要求：(1) 给出两种链表结点struct L1和struct L2的定义（2分）；

(2) 编写 Pair 函数，实现题目要求的功能，允许编写其它函数供 Pair 函数调用（12 分）。

3. 用链表存储一组平面坐标点(x, y)的集合，x和y均为整数。下图所示是3个点的集合(19, 1)，(1, 18)，(14, 0)被存储在链表A的情形：



下图所示是2个点的集合(18, 10)，(10, 10)被存储在链表B的情形：



定义：给定集合A中的一个点a，任取集合B中的一个点b，都可以计算点a与点b的距离 $\text{dist}(a, b)$ ，所有这些距离的最小值称为a到集合B的距离。

今有函数near(A, B, s)，参数A、B指向两个非空链表（表示两组点），参数s为int类型，函数返回新链表C。链表C的结点从A中按顺序复制，但是只选取与B的距离小于s的那些点a。上述链表A和B，执行near(A, B, 15)，结果如下（注意结点的排列次序）：

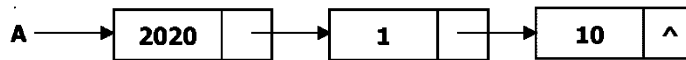


(1) 写出链表结点struct Point的定义；（2分）

(2) 编写距离函数：int dist(int x, int y, struct Point *B)，用于计算点(x, y)到集合B的距离。简单起见，本题采用曼哈顿距离，平面两点a(x1, y1)，b(x2, y2)的距离公式为： $\text{dist}(a, b) = |x1 - x2| + |y1 - y2|$ ；（5分）

(3) 利用(1) (2)完成函数 near，实现题目要求的功能。（7 分）

3. 用链表存储一组字符串($1 \leq \text{串长} \leq 80$)。下图所示是3个字符串"2020", "1", "10"被存储在链表A的情形:

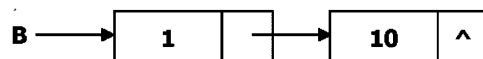


定义: 给定一个字符串s, 指标t(s)定义为s中ASCII码最小的字符的出现次数。

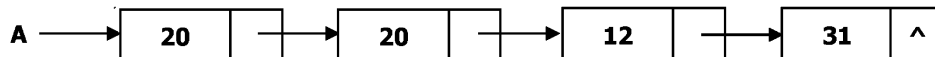
例如, 当s="2020", 最小字符为'0', 出现次数为2, 所以t(s)=2。

根据上述定义, 完成以下问题:

- (1) 规定两个成员名必须为s和next, 写出链表结点struct N的定义; (2分)
- (2) 编写函数: `int t(char *s)`, 计算字符串s的t值; (5分)
- (3) 利用(2), 编写函数ex, 形参是A和n, 功能是将输入的链表A, 按照结点出现顺序将t值正好为n的结点复制出来, 组成一个新链表B并返回, 原链表A不允许做任何修改。例如执行ex(A, 1)后, 所得到的新链表B如下: (7分)

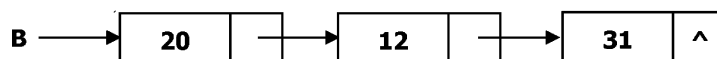


3. 用链表存储一组类型为int的整数。下图所示是4个整数20, 20, 12, 31被存储在链表A的情形:



根据以上描述, 完成以下问题:

- (1) 规定两个成员名必须为d和next, 写出链表结点struct N的定义; (2分)
- (2) 编写函数: `int cnt(struct N* head, int n)`, 返回链表head存储的所有整数中, 出现整数n的次数。例如: `cnt(A, 20)`返回2, `cnt(A, 21)`返回0; (4分)
- (3) 编写函数cp, 形参是A, 功能是按结点出现顺序复制链表A, 但对其中重复的数字, 只保留第一个, 函数返回复制的新链表B, 原链表A不允许做任何修改。例如执行cp(A)后, 得到的新链表B如下: (8分)



提示: 允许调用(2)的cnt函数。

3. 用链表存储一组类型为int的正整数。下图所示是链表A存储4个正整数20，21，12，24的情形：



根据以上描述，完成以下问题：

- (1) 规定两个成员名必须为d和next，写出链表结点struct N的定义；（2分）
- (2) 编写函数：int lastK(struct N* head, int k)，返回链表head的倒数第k个正整数；如果链表为空或者k超出范围，返回-1。例如：lastK(A, 1)返回24，lastK(A, 2021)返回-1，lastK(A, -1)返回-1；（5分）
- (3) 编写函数even，形参是A，功能是按结点出现顺序复制链表A，但是仅仅复制值为偶数的结点，函数返回复制的新链表B，原链表A不允许做任何修改。例如执行B=even(A)后，得到的新链表B如下：（7分）

