**题目描述:一个链表中包含环，请找出该链表的环的入口结点。**

代码(我的):

**A.**

class Solution {

public:

ListNode\* EntryNodeOfLoop(ListNode\* pHead){

if (pHead == NULL)return NULL;

set<ListNode\*>point;

ListNode \*p = pHead;

while (p){

if (point.find(p)==point.end()){

point.insert(p);

p = p->next;

}

else break;

}

return p;

}

};

**B.设置flag标志：**

class Solution {

public:

ListNode\* EntryNodeOfLoop(ListNode\* pHead){

if (pHead == NULL)return NULL;

set<ListNode\*>point;

ListNode \*p = pHead;

bool flag=true;

while (flag&&p!=NULL){

if (point.find(p)==point.end()){

point.insert(p);

p = p->next;

}

else flag = false;

}

return p;

}

};

**总结:**

1.set容器为排序容器，查找find()函数返回迭代器。

2.测试用例中可能含有没有环的链表，则p = p->next;会报错，要加上while (flag&&p!=NULL)。

**C.利用"追及定理"求解**

class Solution {

public:

ListNode\* EntryNodeOfLoop(ListNode\* pHead){

ListNode \*one=pHead,\*two=pHead;

if(pHead==NULL || pHead->next==NULL)return NULL;

do{

one=one->next;

two=two->next->next;

}while(one!=two);

two=pHead;

while(one!=two){

one=one->next;

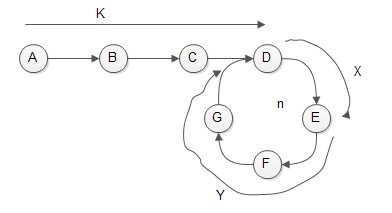
two=two->next; }

return one;

}

};

**方法C原理:**



**首次相遇**

慢指针：K + X + n\*(X+Y) = m;//X+Y 绕环一圈的距离；n 慢指针 总共绕了几圈在环内.

快指针: K+X +N\*(X+Y) = 2\*m;//N为快指针绕过得圈数

联立做差上面两公式:

(N-n)\*(X+Y) = K+X+n\*(X+Y);//这里X+Y 环长是个定值。 假设环长为M.

有：K+X = (N-2\*n)\*M→K+X = Q\*M→K = Q\*M-X

→K = (Q-1)\*M+M-X→K = (Q-1)\*M+Y; 则说明从慢指针从E开始，块指针从A开始(慢指针从A开始，快指针从E开始)速度相同，则在第二次在D点相遇。若k>M则在慢(快)指针在第Q次经过D时与快(慢)指针相遇。