1097. Deduplication on a Linked List (25)

时间限制

300 ms

内存限制

65536 kB

代码长度限制

16000 B

判题程序

Standard

作者

CHEN, Yue

Given a singly linked list L with integer keys, you are supposed to remove the nodes with duplicated absolute values of the keys. That is, for each value K, only the first node of which the value or absolute value of its key equals K will be kept. At the mean time, all the removed nodes must be kept in a separate list. For example, given L being 21→-15→-15→-7→15, you must output 21→-15→-7, and the removed list -15→15.

**Input Specification:**

Each input file contains one test case. For each case, the first line contains the address of the first node, and a positive N (<= 105) which is the total number of nodes. The address of a node is a 5-digit nonnegative integer, and NULL is represented by -1.

Then N lines follow, each describes a node in the format:

*Address Key Next*

where *Address* is the position of the node, *Key* is an integer of which absolute value is no more than 104, and *Next* is the position of the next node.

**Output Specification:**

For each case, output the resulting linked list first, then the removed list. Each node occupies a line, and is printed in the same format as in the input.

**Sample Input:**

00100 5

99999 -7 87654

23854 -15 00000

87654 15 -1

00000 -15 99999

00100 21 23854

**Sample Output:**

00100 21 23854

23854 -15 99999

99999 -7 -1

00000 -15 87654

87654 15 -1

[提交代码](https://www.patest.cn/contests/pat-a-practise/1097)

#include<iostream>

#include<string>

#include<algorithm>

#include<queue>

#include<vector>

#include<sstream>

#include<stack>

#include<map>

#include<cstring>

#include<climits>

#include<cmath>

#include<fstream>

#define MAX 1001

#define LL long long

using namespace std;

#define MAX 10005

struct judge

{

int pos=-1;

};

struct store

{

int data;

int acc = 0;

int next;

int pos;

};

judge judge\_list[10005];

store store\_list[100000];

int main()

{

int startpos, num;

cin >> startpos >> num;

for (int i = 0; i < num; i++)

{

int pos, data, next;

scanf("%d %d %d", &pos, &data, &next);

store\_list[pos].data = data;

store\_list[pos].next = next;

store\_list[pos].acc = 0;

store\_list[pos].pos = pos;

}

int nowpos=startpos;

while (nowpos != -1)

{

int target = abs(store\_list[nowpos].data);

if (judge\_list[target].pos == -1)

judge\_list[target].pos = nowpos;

nowpos = store\_list[nowpos].next;

}

for (int i = 0; i < 10005; i++)

{

if (judge\_list[i].pos != -1)

store\_list[judge\_list[i].pos].acc = 1;

}

nowpos = startpos;

vector<store>selected;

vector<store>droped;

while (nowpos != -1)

{

if (store\_list[nowpos].acc)

{

selected.push\_back(store\_list[nowpos]);

}

else

{

droped.push\_back(store\_list[nowpos]);

}

nowpos = store\_list[nowpos].next;

}

if (!selected.empty())

{

for (int i = 0; i < selected.size() - 1; i++)

{

selected[i].next = selected[i + 1].pos;

}

selected[selected.size() - 1].next = -1;

}

if (!droped.empty())

{

for (int i = 0; i < droped.size() - 1; i++)

{

droped[i].next = droped[i + 1].pos;

}

droped[droped.size() - 1].next = -1;

}

if (!selected.empty())

{

for (int i = 0; i < selected.size(); i++)

{

if (i != selected.size() - 1)

printf("%05d %d %05d\n", selected[i].pos, selected[i].data, selected[i].next);

else

{

printf("%05d %d %d\n", selected[i].pos, selected[i].data, selected[i].next);

}

}

}

if (!droped.empty())

{

for (int i = 0; i < droped.size(); i++)

{

if (i != droped.size() - 1)

printf("%05d %d %05d\n", droped[i].pos, droped[i].data, droped[i].next);

else

{

printf("%05d %d %d\n", droped[i].pos, droped[i].data, droped[i].next);

}

}

}

return 0;

}