**Анализ на решението на задача**

**ИЗГУБЕН ПЪТ**

Неприятна особеност на тези задача е, че градовете се идентифицират с низове със сравнително голяма дължина и всяко обработване на тези низове води до загуба на време. Затова ще заместим тези низове с числа, с помощта на хеширане. Таблицата char ht[2000003][64] e двойно по-голяма отколкото е нужно, за да може да хешираме низовете с колкото може по-малко колизии. За работа с таблицата ни е нужна само функцията int hin(char\* a). Тя запомня зададения като аргумент стринг в хеш-таблицата и връща неговия хеш-номер. Ако низът е вече в таблицата – програмата просто връща хеш-номера му.

Има три възможности за липсващото картонче – да съдържа името на началния град, да е вътрешно за маршрута или да съдържа името на крайния град. Да означим тези три случая с 0, 1 и 2 и да запомним в променливата int c, кой от тях имаме в данните. Нека функцията from\_start() строи в масива int path[] маршрута от града на търговския пътник напред, докато достигне до липсващото картонче, а функцията from\_end() строи в масива int path[] маршрута от крайния градназад, докато достигне до липсващото картонче. В случай 1, след изпълнението на двете функции маршрутът ще бъде построен изцяло. В случай 0 не е необходимо да се изпълнява първата функция, а в случай 2 – втората.

При прочитане на данните, в i-тия ред на масива cards[2000003][2] записваме хеш-номерата на тези градове (0, 1 или 2 на брой), които участват в едно и също картонче с града с хеш-номер i. Ако в картончетата сме намерили само едно, в което е градът с хеш-номер i, тогава на второто място в реда поставяме -1, а ако градът изобщо не се среща в картонче – и двете стойности в реда са -1.

Нека хеш-номерата на началото и края на пътя са hs и hf, а в кой от трите случая сме определяме, като разгледаме какво е записано в реда на началния и крайния град:

c=1; if(cards[hs][0]==-1)c=0;else if(cards[K][0]==-1)c=2;

Така получаваме програма, която решава задачата с линейна сложност.

#include <stdio.h>

#define MAXN 1000001

#define MAXH 2000003

#include <string.h>

int cards[MAXH][2],path[MAXN],K;

char ht[MAXH][64];

int hs,hf;

int hin(char\* a)

{ int n=a[0],l=strlen(a),i=1;

while(a[i]!='\0') n=(26\*n+a[i++])%MAXH;

while(ht[n][0]!='\0'&&strcmp(a,ht[n])!=0) n=(n+1)%MAXH;

if(ht[n][0]=='\0') strcpy(ht[n],a);

return n;

}

void from\_start()

{ path[1]=cards[hs][0];

for(int i=2;i<=K;i++)

{ if(cards[path[i-1]][0]!=path[i-2])

path[i]=cards[path[i-1]][0];

else

if(cards[path[i-1]][1]!=-1)

path[i]=cards[path[i-1]][1];

else break;

}

}

void from\_end()

{ path[K-1]=cards[hf][0];

for(int i=K-2;i>=0;i--)

{ if(cards[path[i+1]][0]!=path[i+2])

path[i]=cards[path[i+1]][0];

else

if(cards[path[i+1]][1]!=-1)

path[i]=cards[path[i+1]][1];

else break;

}

}

int main()

{ int c,i,j,ha,hb; char a[64],b[64],s[64],f[64];

scanf("%d %s %s\n",&K,s,f);

for(i=0;i<MAXH;i++) ht[i][0]='\0';

for(i=0;i<MAXH;i++)cards[i][0]=cards[i][1]=-1;

for(i=1;i<K;i++)

{ scanf("%s %s\n",a,b);

ha=hin(a);hb=hin(b);

if(cards[ha][0]==-1) cards[ha][0]=hb;

else cards[ha][1]=hb;

if(cards[hb][0]==-1) cards[hb][0]=ha;

else cards[hb][1]=ha;

}

hs=hin(s);hf=hin(f);

c=1;

if(cards[hs][0]==-1) c=0;

else if(cards[hf][0]==-1) c=2;

path[0]=hs;path[K]=hf;

if(c!=0) from\_start();

if(c!=2) from\_end();

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

printf("%s\n",ht[path[i]]);

}

*Автор: Красимир Манев*