**Project 13 散列（Hashing）**

学号 2014211116 姓名 骆金参 上交时间 2016-1-1

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 题号 | 程序逻辑(40) | 算法新颖性(20) | 代码规范  (20) | 实验报告(20) | 总分 |
| 得分 |  |  |  |  |  |

1. **问题描述**

对于任意的无序正整数序列，写一程序用哈希排序算法将其排序成按值非递减有序序列。

**2. 算法描述**

**#include<stdio.h>**

**#include<iostream>**

**#include<vector>**

**#include<queue>**

**using namespace std;**

**int main(void) {**

**while( 1 ) {**

**int N;**

**scanf("%d",&N);**

**if( N==0 ) break;**

**int Hash[N]; //Hash数组**

**int Degree[N]; //入度数组**

**vector< vector<int> > G(N); //无向图 二维vector容器**

**vector<int> Ans; //输出序列**

**/\*输入hash序列\*/**

**for (int i = 0; i < N; i++) {**

**scanf("%d",&Hash[i]);**

**if (Hash[i] > 0) Degree[i] = 0;**

**else Degree[i] = -1;**

**//如果小于0，入度记为-1，表示没有元素**

**}**

**/\*计算入度并建立无向图\*/**

**for (int i = 0; i < N; i++){**

**if (Hash[i] < 0) continue;**

**int curPos = i; //当前坐标**

**int hashPos = Hash[i] % N; //Hash后的坐标**

**Degree[i] = (curPos - hashPos + N) % N; //计算入度，也就是冲突的次数**

**for (int j = 0; j < Degree[i]; j++)**

**G[(hashPos + j + N) % N].push\_back(i);**

**}**

**/\*拓扑排序\*/**

**typedef pair<int, int> PAIR;**

**priority\_queue<PAIR, vector< PAIR >, greater< PAIR > > q; //优先队列**

**for (int i = 0; i < N; i++) {**

**if (Degree[i] == 0) {**

**q.push(PAIR(Hash[i], i));**

**}**

**}**

**while ( !q.empty() ) {**

**PAIR p = q.top(); //每次取出当前入度为0的顶点中Key最小的**

**int V = p.second; //second为顶点**

**Ans.push\_back(p.first); //first为该顶点的Key**

**q.pop();**

**for (int i = 0; i < G[V].size(); i++) //扫描关联顶点，入度处理**

**if (--Degree[G[V][i]] == 0)**

**q.push(PAIR(Hash[G[V][i]], G[V][i]));**

**}**

**/\*输出\*/**

**printf("%d",Ans[0]);**

**for (int i = 1; i < Ans.size(); i++)**

**printf(" %d",Ans[i]);**

**printf("\n");**

**}**

**return 0;**

**}**

**3.测试结果**



**4. 分析与评论**

简单的哈希法包括余数法和折叠等，实验即折叠法，以输入的N为标准，将数据分为若干堆，再连接起来。