|  |  |  |  |
| --- | --- | --- | --- |
|  | | | |
| Федеральное государственное бюджетное  образовательное учреждение высшего образования FPMI_ngtu_neti_rgb_polya«Новосибирский государственный технический университет» | | | |
|  | | | |
| Кафедра прикладной математики | | | |
| Практическое задание № 5 | | | |
| по дисциплине «Структуры данных и алгоритмы» | | | |
| **«Алгоритмы сортировки»** | | | |
|  | | | |
|  | Группа | ПМ-21 | |
| Бригада 7 | Егупов иван | |
|  | Щербакова Мария | |
| Сагыдыев Данияр | |
|  | |
|  | |
| Преподаватель | ХИЦЕНКО ВАЛЕНТИНА ПАВЛОВНА | |
| Дата | 20.05. | |
| Новосибирск,2023 | | | |

1. **Условие задачи**

Дополнить таблицу, построенную в практическом задании «ТАБЛИЦЫ» (вариант 2в), информацией о цене изделия. Цену изделия брать из таблицы – прейскурант, элемент которой содержит: шифр изделия, цена (за штуку). Эта таблица упорядочена по возрастанию шифров изделий. Упорядочить преобразованную таблицу по новому ключу – количество изделий методом сортировки с использованием струтуры дерево.

Для поиска элементов в таблице использовать последовательный поиск.

**2. Анализ задачи**

Входные данные:

Table = {((eli = (keyi, namei, producedi)),n)| keyi ∈ G или ∅, namei ∈ M, , producedi ∈ N,i<=n}

G = {gi|g i∈ {‘a’,’b’… ‘z’,’1’,’2’…’0’}, i = 1,8}

M = {mi|m i∈ {‘a’,’b’… ‘z’}, i = 1,10}

Price = {((keyi,pricei),k)|keyi ∈ G, pricei ∈ N, i<=k}

Выходные данные:

Table1 = {(eli = (keyi, namei, producedi, pricei)),n)| keyi ∈ G, namei ∈ M, , producedi ∈ N,pricei ∈ N, ∀ i: producedi <= producedi+1, i <=20}

ncal, ncom,ns ∈ N

Решение:

3.1 Построить пирамиду

**при** i = крайний элемент с предпоследнего уровня дерева

**повторять**

**при** j = правый потомок i и v = левый потомок i

**повторять** *если* j>i, то обменять местами j и i, j = правый потомок i  
 v = левый потомок i, *иначе* обменять местами v и i, j = правый  
 потомок i, v = левый потомок i

**пока** ∃j и ∃v

i = следующий элемент с текущего уровня, если элементы на уровне  
 закончились, то перейти на уровень выше

**пока** ∃i

3.2 Сортировка с использованием пирамиды

**при** x = последний элемент неотсортированной части , Table - построенная пирамида

**повторять**: обменять метами x и первый элемент Table , положить последний элемент в отсортированную часть, построить пирамиду для i = первый элемент , x = последний элемент неотсортированной части

**пока** ∃x

3.3 Добавление информации в таблицу

**при** i = 0;

**повторять** если Table[i]key ≠ ∅, то

**при** j = 0, flag = 1

**повторять** если Table[i]key = Price[j]key  , то добавить информацию о цене,  
 flag = 0, иначе j = j +1

**пока** flag = 1 И j < k

i = i+1

**пока** i<n

Подзадачи

*Buildtree*– построение (балансировка) бинарного дерева до пирамиды

Входные данные: бинарное дерево, индекс максимального элемента, общее количество элементов дерева

Выходные данные: Сбалансированное дерево

*pyramidsort–* сортировка неупорядоченной таблицы методом пирамидальной сортировки

Входные данные: Неупорядоченная таблица, общее количество элементов таблицы

Выходные данные: отсортированная по возрастанию нового ключа ( количество изделий) таблица.

*add\_price* – добавление в таблицу Table информации о цене изделий из таблицы Price

*Входные данные*: Table и Price, количество элементов в обеих таблицах

*Выходные данные*: Table1

**3.Структуры данных**

Входные данные

*Внешнее представление:*

В файле Table.txt содержатся элементы, состоящие из индекса элемента в таблице, вычисленного для него хэш-значения, шифра изделия (8-символьный код), наименования изделия, количества штук. Каждая состовляющая элемента написана через пробел. Элемент – новая строка.

В файле Price.txt содержатся элементы, состоящие из шифра изделия таблицы Table, а также стоимости одной еденицы изделия. Каждая составляющая написана через пробел. Элемент – новая строка.

*Внутреннее представление:*

Элемент Последовательности Table представлен следующей структурой

struct Elem

{

char key[K];

char name[B];

int produced;

int price;

};

const char K = 9; const char B = 11;

Последовательность Table представлена одноименной струтурой

struct Table

{

struct Elem[N];

int n;

}

const cahr N = 20

ВЫХОДНЫЕ ДАННЫЕ

*Внешнее представление:*

С новой строки последовательность, упорядоченная по возрастанию количества произведенных езделий, каждая составляющая написана через пробел: количество изделий, шифр изделия, наименование изделия и стоимость изделия. Следующий элемент – новая строка.

*Внутреннее представление:* аналогично входным данным

**4.Алгоритм решения задачи**



4.1 *Подзадача добавления информации о цене*

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

{

bool flag = true;

if (элемент таблицы не пустой)

{

for (int j = 0; j < k && flag; j++) if (ключи равны)

{

elem[i].price += pr[j].price; flag = false;

}

}

}

*4.2 Подзадача построения/балансировки пирамиды*

int max = i (индекс, от которого строим пирамиду)

int j = 0;

do

{

ncal++;

j = max;

int l = 2 \* j + 1;

int r = 2 \* j + 2;

if (если существует правый потомок и потомки больше корня)

{

if (правый потомок больше левого)

max = r;

else

max = l;

}

else

if (левый потомок существует и он больше корня)

max = l;

swap(elem[j], elem[max]);

} while (max != j);

*4.3 Подаздача Сортировки пирамидой*

for (int i = n / 2; i >= 0; i--)

Построить Пирамиду от i

for (int i = n - 1; i >= 0; i--)

{

ns++;

swap(elem[0], elem[i]);

Построить Пирамиду от корня

}

5.Структура программы

void add\_price(Elem \*elem, Price\* pr, int n,int k) Назначение: добавление в таблицу Table информации о цене из таблицы Price

Elem \*elem – указатель на таблицу Table

Price \*pr – указатель на таблицу Price

int n – размер таблицы Table

int k – размер таблицы Price

void buildTree (Elem \*elem,int i,int n,int &ncal,int &ncom,int &ns) Назначение: построение или балансировка пирамиды, используемой при сортировке

Elem \*elem – указатель на массив, которым представлено бинарное дерево

int i – индекс корня, от которого производим балансировку

int n – общее число элементов в массиве

&int ncal – указатель на переменную, хранящую число просмотров

&int ncom – указатель на переменную, хранящую число сравнений

&int ns – указатель на переменную, хранящую число перемещений

void pyramidSort(Elem \*elem, int n, int& ncal, int& ncom, int& ns) Назначение: пирамидальная сортировка таблицы

Elem \*elem – указатель на массив,который сортируем

int n – общее число элементов в массиве

6.Текст программы

Заголовочный файл “Header.h”

#pragma once

#include <stdio.h>

#include <string.h>

#include <iostream>

using namespace std;

const int K = 9;

const int B = 11;

const int N = 40;

struct Elem

{

int hash;

char key[K];

char name[B];

int produced = 0;

int price = 0;

};

struct Table

{

struct Elem elem[N]{};

int n;

};

struct Price

{

char key[K];

int price;

};

void add\_price(Elem\* elem, Price\* pr, int n, int k)

{

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

{

bool flag = true;

if (strlen(elem[i].key) != 0 )

for (int j = 0; j < k && flag; j++)

if (strcmp(elem[i].key, pr[j].key) == 0)

{

elem[i].price += pr[j].price;

flag = false;

}

}

}

void buildTree (Elem \*elem,int i,int n,int \*ncal,int \*ncom,int \*ns)

{

int max = i;

int j = 0;

do

{

(\*ncal)++;

j = max;

int l = 2 \* j + 1;

int r = 2 \* j + 2;

(\*ncom)++;

if (r <= n && (elem[r].produced > elem[j].produced || elem[l].produced > elem[j].produced))

{

if (elem[r].produced > elem[l].produced)

max = r;

else

max = l;

}

else

{

(\*ncom)++;

if (l <= n && elem[l].produced > elem[j].produced) max = l;

}

if (max != j)

{

(\*ns)++;

swap(elem[j], elem[max]);

}

} while (max != j);

}

void pyramidSort(Table \*T ,int \*ncal, int \*ncom, int \*ns)

{

for (int i = T->n / 2; i >= 0; i--)

buildTree(T->elem, i, T->n - 1, ncal, ncom, ns);

for (int i = T->n - 1; i >= 0; i--)

{

(\*ns)++;

swap(T->elem[0],T->elem[i]);

buildTree(T->elem, 0, i - 1, ncal, ncom, ns);

}

}

Основная программа Source.cpp

#include "Header.h"

int main()

{

Table T;

T.n = N;

FILE\* tab;

fopen\_s(&tab, "Table.txt", "r");

while (!feof(tab))

{

int p;

fscanf\_s(tab, "%d", &p);

fscanf\_s(tab, "%d%s%s%d",&T.elem[p].hash,& T.elem[p].key, K, &T.elem[p].name, B, &T.elem[p].produced);

}

fclose(tab);

fopen\_s(&tab, "Price.txt", "r");

Price Po[N];

int i = 0;

while (!feof(tab))

{

Price el;

fscanf\_s(tab, "%s%d", &el.key, K, &el.price);

Po[i] = el;

i++;

}

fclose(tab);

add\_price(T.elem, Po, T.n,i+1);

int ncal = 0; // просмотры

int ncom = 0; // сравнения

int ns = 0; // перемещения

pyramidSort(&T, &ncal, &ncom, &ns);

fopen\_s(&tab, "Table1.txt", "w");

for (int i = 0; i < T.n; i++)

if (strlen(T.elem[i].key)!=0)

fprintf\_s(tab,"%d %s %s %d\n", T.elem[i].produced,T.elem[i].key, T.elem[i].name, T.elem[i].price);

fprintf\_s(tab,"\nncal: %d ncom: %d ns: %d", ncal, ncom, ns);

fclose(tab);

return 1;

}

7.Набор тестов

|  |  |  |
| --- | --- | --- |
| Входные данные | Выходные данные | Назначение |
| Table.txt:  8 8 bvyewttt OPew 32  11 11 Vbhfyewp LOOOP 1  12 12 Mnpi652x qwert 100  13 13 ywetr234 Pirw 59  14 14 8743610b Kour 40  Price.txt:  bvyewttt 67  Vbhfyewp 78  Mnpi652x 56  ywetr234 33  8743610b 12 | 1 Vbhfyewp LOOOP 78  32 bvyewttt OPew 67  40 8743610b Kour 12  59 ywetr234 Pirw 33  100 Mnpi652x qwert 56  ncal: 46 ncom: 77 ns: 35 | В таблице было 5 ненулевых элементов, к каждому была добавлена информация о цене. |
| Table.txt:  3 3 otr56BN3 Dima 40  4 4 trytvvb2 Oropfvw 5222  6 6 netwyr90 Trakimus 10000  12 12 bvry89YR Zavik 120  13 13 bvrepw00 BVHewe 13  15 15 teyyvve2 Porvbed 562  17 17 yrue46PO Vanya 100  19 19 pte6736v Poland 89  Price.txt:  otr56BN3 99  bvry89YR 13  yrue46PO 53  pte6736v 899  pwpwpwpw 89 | 13 bvrepw00 BVHewe 0  40 otr56BN3 Dima 99  89 pte6736v Poland 899  100 yrue46PO Vanya 53  120 bvry89YR Zavik 13  562 teyyvve2 Porvbed 0  5222 trytvvb2 Oropfvw 0  10000 netwyr90 Trakimus 0  ncal: 56 ncom: 88 ns: 45 | В Table содержалось 8 элементов,в Price содержалось 5 элементов,но шифр одного из них не соответствовал ни одному элементу Table |
| Table.txt  3 3 otr56BN3 Dima 0  4 4 trytvvb2 Oropfvw 0  6 6 netwyr90 Trakimus 0  12 12 bvry89YR Zavik 0  13 13 bvrepw00 BVHewe 0  15 15 teyyvve2 Porvbed 0  17 17 yrue46PO Vanya 0  19 19 pte6736v Poland 0  Price.txt  otr56BN3 99  bvry89YR 13  yrue46PO 53  pte6736v 899  trytvvb2 67  bvrepw00 66  teyyvve2 777  netwyr90 5 | 0 otr56BN3 Dima 99  0 trytvvb2 Oropfvw 67  0 netwyr90 Trakimus 5  0 bvry89YR Zavik 13  0 bvrepw00 BVHewe 66  0 teyyvve2 Porvbed 777  0 yrue46PO Vanya 53  0 pte6736v Poland 899  ncal: 31 ncom: 62 ns: 20 | Всем элементам поле produced было присвоено ноль, таким образом исходная таблица, включая нулевые строки, была уже упорядочена |
| Table.txt  0 17 rfgY67Ha lamp 13  1 19 xxx0KRt3 tablecl 14  2 2 Yyyy56UU dress 2  3 18 vbdngveg ooooo 44  4 4 678jkl11 cradle 11  5 15 vbneeern ppwew 98  6 6 etwytwyw uurur 89  7 14 xvbeweww wwwwe 78  8 8 1Uki89fD hammock 8  9 9 WsR43Kig sofa 1  10 9 98Mi34h4 stool 15  11 17 ewewtyew vbvbv 90  12 12 mkt78GY1 cabinet 6  13 13 234hyrt8 carpet 7  14 14 XXX0krt3 mirror 9  15 15 ukII543L refrig 5  16 14 Nun777FS wardro 10  17 15 yhbvGHas cot 12  18 18 1as45hr8 booksh 4  19 19 asDF0lkP stool 3  Price.txt  rfgY67Ha 7  xxx0KRt3 5  Yyyy56UU 23  vbdngveg 34  678jkl11 54  vbneeern 45  etwytwyw 45  xvbeweww 43  1Uki89fD 34  WsR43Kig 54  98Mi34h4 89  ewewtyew 12  mkt78GY1 32  234hyrt8 44  XXX0krt3 55  ukII543L 78  Nun777FS 1  yhbvGHas 2  1as45hr8 4  asDF0lkP 5 | 1 WsR43Kig sofa 54  2 Yyyy56UU dress 23  3 asDF0lkP stool 5  4 1as45hr8 booksh 4  5 ukII543L refrig 78  6 mkt78GY1 cabinet 32  7 234hyrt8 carpet 44  8 1Uki89fD hammock 34  9 XXX0krt3 mirror 55  10 Nun777FS wardro 1  11 678jkl11 cradle 54  12 yhbvGHas cot 2  13 rfgY67Ha lamp 7  14 xxx0KRt3 tablecl 5  15 98Mi34h4 stool 89  44 vbdngveg ooooo 34  78 xvbeweww wwwwe 43  89 etwytwyw uurur 45  90 ewewtyew vbvbv 12  98 vbneeern ppwew 45  ncal: 83 ncom: 117 ns: 72 | Хэш таблица была полна |
| Table.txt  0 17 rfgY67Ha lamp 20  1 19 xxx0KRt3 tablecl 19  2 2 Yyyy56UU dress 18  3 18 vbdngveg ooooo 17  4 4 678jkl11 cradle 16  5 15 vbneeern ppwew 15  6 6 etwytwyw uurur 14  7 14 xvbeweww wwwwe 13  8 8 1Uki89fD hammock 12  9 9 WsR43Kig sofa 11  10 9 98Mi34h4 stool 10  11 17 ewewtyew vbvbv 9  12 12 mkt78GY1 cabinet 8  13 13 234hyrt8 carpet 7  14 14 XXX0krt3 mirror 6  15 15 ukII543L refrig 5  16 14 Nun777FS wardro 4  17 15 yhbvGHas cot 3  18 18 1as45hr8 booksh 2  19 19 asDF0lkP stool 1  Price.txt  rfgY67Ha 7  xxx0KRt3 5  Yyyy56UU 23  vbdngveg 34  678jkl11 54  vbneeern 45  etwytwyw 45  xvbeweww 43  1Uki89fD 34  WsR43Kig 54  98Mi34h4 89  ewewtyew 12  mkt78GY1 32  234hyrt8 44  XXX0krt3 55  ukII543L 78  Nun777FS 1  yhbvGHas 2  1as45hr8 4  asDF0lkP 5 | 1 asDF0lkP stool 5  2 1as45hr8 booksh 4  3 yhbvGHas cot 2  4 Nun777FS wardro 1  5 ukII543L refrig 78  6 XXX0krt3 mirror 55  7 234hyrt8 carpet 44  8 mkt78GY1 cabinet 32  9 ewewtyew vbvbv 12  10 98Mi34h4 stool 89  11 WsR43Kig sofa 54  12 1Uki89fD hammock 34  13 xvbeweww wwwwe 43  14 etwytwyw uurur 45  15 vbneeern ppwew 45  16 678jkl11 cradle 54  17 vbdngveg ooooo 34  18 Yyyy56UU dress 23  19 xxx0KRt3 tablecl 5  20 rfgY67Ha lamp 7  ncal: 74 ncom: 106 ns: 63 | Таблица Table упорядочена в обратном порядке |