|  |  |  |
| --- | --- | --- |
|  | | |
| Федеральное государственное бюджетное  образовательное учреждение высшего образования FPMI_ngtu_neti_rgb_polya«Новосибирский государственный технический университет» | | |
|  | | |
| Кафедра прикладной математики | | |
| Практическое задание № 4 | | |
| по дисциплине «Структуры данных и алгоритмы» | | |
| **Управление таблицами** | | |
|  | | |
|  | Бригада 3 | лезнёв артём |
| Группа ПМ-15 | черновская мария |
| Вариант 2а | зуев андрей |
|  |  |
|  |  |
|  |  |
| Преподаватели | Тракимус Юрий Викторович |
|  |  |
| Новосибирск, 2022 | | |

1. **Условие задачи**

В файле WORK содержатся результаты работы цеха за день. Элемент файла включает: шифр изделия (8-символьный код), наименование изделия, количество (штук). Построить таблицу, содержащую результаты работы за день, считая ключом шифр изделия. Элемент таблицы имеет ту же структуру, что и элемент файла. Содержащаяся в файле информация с равными ключами должны быть помещена в таблицу один раз с общим количеством штук изделия. Организовать таблицу как: неупорядоченную.

1. **Анализ задачи**

Входные данные: **WORK.txt** - таблица с результатами работы цеха за день.

Выходные данные: сортированная таблица с данными по каждому изделию (шифр, название, количество изготовленных штук).

Структуру входной таблицы с данными определим следующим образом:

|  |  |  |
| --- | --- | --- |
| **Шифр** | **Название** | **Количество** |
| 00000001 | Название1 | 30 |
| 00000002 | Название2 | 20 |
| 00000001 | Название1 | 40 |

Итоговую таблицу будем сортировать по 8-значному коду.

Обработка входного файла:

1. Считываем код, название и количество произведенных штук одного изделия.

2. Проверяем код на уникальность.

3. Если код уникален, добавляем данные изделия в таблицу, иначе при повторном коде добавляем к количеству произведенных штук изделия количество при повторном коде.

Сортировка таблицы происходит по следующему алгоритму:

Функция **insert\_or\_assign** добавляет новый элемент **item** в таблицу или возвращает указатель на старый, если он уже есть в таблице.

1. Считываются данные с входного файла.

2. Происходит проверка таблицы на полноту.

2. Далее идет поиск элемента с таким же ключом.

3. Если поиск не удался элемент помещается в массив.

4. Иначе размер таблицы уменьшается на единицу.

5. К количеству изготовленных штук изделия прибавляется текущее количество. (Если элемент уникальный, количество изготовленных изделий такое же как в входном файле. Если не уникальный, то количества изготовленных штук изделия складываются).

Таблица после подсчета всех голосов:

|  |  |  |
| --- | --- | --- |
| **Шифр** | **Название** | **Количество** |
| 00000001 | Название1 | 70 |
| 00000002 | Название2 | 20 |

После сортировки таблица выводится в консоль.

1. **Текст программы**
2. #include <stdio.h>
3. #include <conio.h>
4. #include <windows.h>
5. const UCHAR N = 20, K = 9, Nn = 100;
6. struct elem
7. {
8. UINT quantity = 0;
9. char key[K]{ }, name[Nn]{ };
10. };
11. struct table
12. {
13. UCHAR n = 0;
14. elem elems[N]{};
15. void insert(elem& e)
16. {
17. UCHAR p = binarySearch(e.key);
18. if (!strcmp(elems[p].key, e.key))
19. elems[p].quantity += e.quantity;
20. else
21. {
22. shift(p);
23. elems[p] = e;
24. }
25. }
26. int binarySearch(char\* key)
27. {
28. UCHAR l = 0, r = n;
29. for (; l < r; )
30. {
31. UCHAR m = (l + r) / 2;
32. if (strcmp(elems[m].key, key) < 0)
33. l = m + 1;
34. else
35. r = m;
36. }
37. return l;
38. }
39. void shift(UCHAR p)
40. {
41. n++;
42. for (CHAR i = n - 1; i >= p; i--)
43. elems[i + 1] = elems[i];
44. }
45. void deleteF(char\* key)
46. {
47. UCHAR p = binarySearch(key);
48. if (strcmp(key, elems[p].key)) p = 0;
49. if (p)
50. {
51. for (UCHAR i = p; i < n; i++)
52. elems[i] = elems[i + 1];
53. n--;
54. }
55. else printf\_s("Элемента в таблице нет.\n");
56. }
57. void input(FILE\* f)
58. {
59. elem e;
60. fopen\_s(&f, "WORK.txt", "r");
61. if (f == NULL)
62. printf("Ошибка открытия файла\n");
63. else
64. {
65. for (; fscanf\_s(f, "%s", e.key, \_countof(e.key)) != EOF; insert(e))
66. fscanf\_s(f, "%s%u", e.name, \_countof(e.name), &e.quantity);
67. fclose(f);
68. }
69. }
70. void output(FILE\* f)
71. {
72. fopen\_s(&f, "WORKOUT.txt", "w");
73. if (f == NULL)
74. printf("Ошибка открытия файла\n");
75. else
76. {
77. for (UCHAR i = 0; i < n; i++)
78. fprintf\_s(f, "%s %s %5u\n", elems[i].key, elems[i].name, elems[i].quantity);
79. fclose(f);
80. }
81. }
82. };
83. int main()
84. {
85. UINT outCp = GetConsoleOutputCP();
86. SetConsoleOutputCP(1251);
87. table\* t = new table;
88. UCHAR s = 0;
89. FILE\* f = NULL;
90. t->input(f);
91. s = getchar();
92. t->deleteF(t->elems[s].key);
93. if (f == NULL)
94. t->output(f);
95. SetConsoleOutputCP(outCp);
96. return 0 \* \_getch();
97. }
98. **Набор тестов**

|  |  |  |
| --- | --- | --- |
| **№** | **Входные данные** | **Назначение** |
| 1 | 11111111 Toy 20  98874300 Book 100  87654321 Dress 209  33091265 Pencil 41 | Проверка программы на все уникальные коды. |
| 2 | 32345235 Short 899  00120034 Computer 34  67088176 Table 125  33091265 Window 211  00120034 Computer 111  32345235 Short 96  67088176 Table 11  33091265 Window 1  33091265 Window 78 | Проверка программы на повторяющиеся коды всех изделий. |
| 3 | 00000001 Cup 3276  00120034 Knife 3  00120034 Knife 3  10203040 Spoon 13098  20040030 Fork 2001  00120034 Knife 3  33557799 Plate 111  00120034 Knife 3  00120034 Knife 3 | Проверка программы на один много раз повторяющийся код |

1. **Результаты работы программы**

|  |  |
| --- | --- |
| **№** | **Ввод/вывод программы** |
| 1 | Шифр Название Количество  11111111 Toy 20  98874300 Book 100  87654321 Dress 209  33091265 Pencil 41 |
| 2 | Шифр Название Количество  32345235 Short 995  00120034 Computer 145  67088176 Table 136  33091265 Window 290 |
| 3 | Шифр Название Количество  00000001 Cup 3276  00120034 Knife 15  10203040 Spoon 13098  20040030 Fork 2001  33557799 Plate 111 |