**Лабораторная работа 2**

***Общее задание***: Создать таблицу, содержащую не менее 40-ка записей (тип – запись с вариантами). Упорядочить данные в ней по возрастанию ключей, где ключ – любое невариантное поле (по выбору программиста), используя: а) саму таблицу, б) массив ключей (возможность добавления и удаления записей в ручном режиме обязательна)

Сравнить различные алгоритмы сортировки массива при использовании таблиц записей с большим числом полей и таблиц ключей. Оценить эффективность (по времени и по используемому объему памяти) при различной реализации программы, то есть в случаях (1) и (2).

**Таблица**: список абонентов, содержащий фамилию, имя, телефон, адрес, статус (личный – дата рождения: день, месяц, год; служебный – должность, организация).

***Техническое задание***

Осуществляются операции (добавление записи, удаление записи и т.д.) с таблицей записей и таблицей ключей, представленными в памяти в виде массива записей. Найти.

**Входные данные**: бинарный файл, представляющий таблицу записей с вариантной частью.

**Выходные данные**: список записей, удовлетворяющих условиям (все друзья, которых необходимо поздравить с днем рождения в ближайшую неделю). Две таблицы записей, отсортированные по ключам. Отсортированные таблицы ключей. Время работы для различных сортировок.

**Возможные ошибки**:

* Открытие несуществующего файла

Сообщение:

{ Unable to load! }

* Открытие файла с некорректным форматом данных

|  |  |  |  |
| --- | --- | --- | --- |
| Содержимое файла | Результат | | Пояснение |
|  | { Unable to load! } | | Пустой файл |
| 2  Suzuki SX4  Black  Japan  1304000  0  4 | { Unable to load! } | | Ожидается 2 элемента, а записан 1 |
| qaaq | { Unable to load! } | | Ожидается целое, подана строка |
| Команда | | Пример результата | |
| 1 | | { Unable to load! } | |
| 2 | | { Successfully saved } | |
| 3 | | { Added new record } | |
| 4 | | { Removed record } | |
| 5 | | { Sorted table } | |
| 6 | | Table is empty. | |
| 7 | | { Created key table } | |
| 8 | | { Sorted key table } | |
| 9 | | Key table is empty. | |
| 10 | | Table is empty. | |
| 11 | | [ Record 41 ]  Car model: Kia Soul  Country: Korea  Price: 1384000  Color: Red  State: New  Warranty (years): 9 | |
| 12 | | { Table sorting (qsort) = 9240648 }  { Key table sorting (qsort) = 3834740 }  { Table sorting (bubble sort) = 64941998 }  { Key table sorting (bubble sort) = 41243816 }  { Table size = 7392 B }  { Key table size = 336 B } | |

**Описание структур данных**

**Абонент**

typedef struct

{

char surname[30];

char name[20];

int phone;

char adres[40];

enum

{

personal,

work

}st;

union

{

struct data dofb;

struct office ofc;

} type;

} listp;

**Ключ**

typedef struct

{

int index\_src;

char surname[30];

} key\_listp;

**Ключ – статус «личный»**

struct data

{

int d;

int m;

int y;

};

**Ключ – статус «служебный»**

struct office

{

char position[20];

char organization[20];

};

**Время выполнения**

|  |  |  |
| --- | --- | --- |
| Метод сортировки | Сортируемая таблица | Количество тиков |
| Быстрая | Абонентов |  |
| Быстрая | Ключей |  |
| Шейкерная | Абонентов |  |
| Шейкерная | Ключей |  |

**Количество потребляемой памяти**

Таблица абонентов (63 записи) Таблица ключей (42 записи)

7392 B 336 B

**Вывод**

Таблица ключей сортируется примерно в 5 раз быстрее, чем исходная таблица;

Быстрая сортировка примерно в 3 раза быстрее, чем сортировка пузырьком.

**Ответы на вопросы**

*Как выделяется память под вариантную часть записи?* – Выделяется область памяти, равная размеру максимального по длине поля вариантной части.

*Что будет, если в вариантную часть ввести данные, несоответствующие описанным?* – Тип данных в вариантной части при компиляции не проверяется, все проверки необходимо осуществлять самостоятельно.

*Кто должен следить за правильностью выполнения операций с вариантной частью записи?* – Разработчик.

*Что представляет собой таблица ключей, зачем она нужна?* – Таблица ключей содержит индекс элемента в исходной таблице и выбранный ключ. За счет небольших дополнительных затрат памяти позволяет ускорить процесс поиска и сортировки элементов исходной таблицы.

*В каких случаях эффективнее обрабатывать данные в самой таблице, а когда – использовать таблицу ключей?* – Использовать таблицу ключей эффективнее в случае большого количества записей или большого размера памяти, необходимой для хранения каждой записи.

*Какие способы сортировки предпочтительнее для обработки таблиц и почему?* – При сортировке таблиц эффективнее использовать таблицу ключей. При этом быстрая сортировка работает быстрее, чем сортировка пузырьком.