|  |  |
| --- | --- |
|  | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ \_\_\_\_\_\_\_\_Информатика и системы управления\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

КАФЕДРА \_\_\_\_\_\_\_Программное обеспечение ЭВМ и информационные технологии\_\_\_\_\_\_\_\_

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ ПО ПРЕДМЕТУ "ТИПЫ И СТРУКТУРЫ ДАННЫХ" №2**

Студент\_Золотухин Алексей Вячеславович\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*фамилия, имя, отчество*

Группа\_\_\_\_\_\_ИУ7-34Б\_\_\_\_\_\_\_\_\_

Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_Золотухин А. В.\_\_\_\_

*подпись, дата фамилия, и.о.*

Принял \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_Силантьева А. В. \_

*подпись, дата фамилия, и.о.*

Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Москва, 2021**

**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).

Ввести список абонентов, содержащий фамилию, имя, телефон, адрес, статус (личный – дата рождения: день, месяц, год;

служебный – должность, организация). Найти всех друзей, которых

необходимо поздравить с днем рождения в ближайшую неделю.

**Описание программы**

Данная программа предназначена для работы со списком машин и представляет собой консольное приложение со следующими возможными операциями:

* Добавление строки в список.
* Удаление строки из списка.
* Поиск всех друзей, которых необходимо поздравить с днем рождения в ближайшую неделю.
* Вывод отсортированной таблицы ключей.
* Вывод отсортированного списка абонентов.
* Вывод отсортированного списка абонентов с применением отсортированной таблицы ключей.
* Вывод временной разницы между таблицы ключей и исходного списка.
* Вывод временной разницы между сортировками разными способами.
* Вывод исходного списка абонентов.

Требования к функциональным характеристикам: программа должна считывать данные таблицы из файла, при ошибке чтения программа не выполняет никаких действий.   
 Для коррекиной работы программе требуется файл data.txt  
 В программе должно быть реализовано меню с возможностью выбора разных вариантов работы с таблицей.   
   
 При неправильном вводе пункта меню программа завершается.   
  
 При добавлении записи в таблицу, значение каждого поля новой записи должно вводиться отдельно.   
  
 Поле “Фамилия” не превышает 15 символов, “Имя” -- 15, “Номер телефона” -- 15, "Адрес" - 55, "Должность" -- 30, "Организация" -- 40,остальные поля целочисленные.   
  
 После ввода новая запись добавляется в таблицу.

В таблице может быть не больше 60 абонентов.   
   
 Для удаления записи из таблицы должен быть введен номер абонента удаляемой записи. После ввода запись удаляется из таблицы.  
  
 При выводе всех данных таблица выводится в удобочитаемом формате.   
  
 При поиске вводится дата в формате дд.мм.гггг.   
  
 После ввода на экран выводятся все подходящие записи если ничего не вывелось значит ничего не нашлось.

При сортировке таблица выводится в удобочитаемом виде в лексикографичексом порядке поля “Фамилия”. Также должна быть реализована сортировка таблицы с помощью дополнительной таблицы ключей.   
  
 При неправильном вводе каких-либо параметров пользователь возвращается к меню. После проведения операций пользователь выбирает, продолжить работу с программой или завершить.

Требования к надежности: программа должна воспринимать любой ввод и не завершаться в аварийном режиме.   
  
 Если в символьное поле таблицы ввели больше 15 символов, то пользователь переходит к меню.   
  
 Если в числовое поле введено не целое положительное число (кроме срока гарантии - число вещественного типа), то пользователь переходит к меню.

Требования к эксплуатации: программа должна запускаться с лицензионной ОС, которая поддерживает исполняемые файлы формата «.exe». Программа не требует проведения каких-либо видов обслуживания.

**Исходные структуры данных**

Таблица, содержащая информацию об абонетах

phone\_t table[60];

Таблица ключей:

key\_t k[60];

Запись содержащая общую информацию об абоненте

#define SURNAMELEN 15

#define NAMELEN 15

#define NUMBERLEN 15

#define POSTLEN 30

#define ORGLEN 40

#define ADRESSLEN 55

#define STATUSLEN 10

typedef struct

{

char surname[SURNAMELEN];

char name[NAMELEN];

char number[NUMBERLEN];

char adress[ADRESSLEN];

char status;

status\_t status\_u;

} phone\_t;

Запись-ключ

typedef struct

{

size\_t prev\_i;

char surname[SURNAMELEN];

} key\_t;

Запись с персональной информацией

typedef struct

{

int day;

int month;

int year;

} personal\_t;

Запись с рабочей информацией

typedef struct

{

char post[POSTLEN];

char organization[ORGLEN];

} official\_t;

Объединение для вариантной информации

typedef union

{

personal\_t p;

official\_t o;

} status\_t;

**Сравнение различных методов сортировки**

В данной лабораторной работе мною были рассмотрены и сравнены два вида сортировки – пузырьком и «шейкером».

Ключ (поле для сортировки) – фамилии абонентов.

**Анализ алгоритмов сортировки**

Замеры времени (в микросекундах):



*(Замеры были произведены на списке, содержащем 40 элементов, при 1005-кратном повторении – в таблице приведены усреднённые результаты).*

|  |  |  |
| --- | --- | --- |
|  | Время сортировки таблицы ключей | Время сортировки всей таблицы |
| Пузырёк | 11 | 19 |
| Шейкер | 5 | 14 |

Сортировка таблицы ключей эффективнее сортировки всей таблицы на 42% при сортировке пузырьком и на 64% эффективнее при двунаправленной сортировке.

Время сортировки таблицы ключей шейкером эффективнее чем пузырьком на 54%.

Время сортировки исходной таблицы шейкером эффективнее чем пузырьком на 26%.

Однако метод двунаправленной сортировки работает медленнее на больших объёмах данных, поэтому требует большего количества времени, нежели сортировка пузырьком.

Минимальный объём памяти, необходимый для содержания всей таблицы: 171 \* 40 = 6840 байт. Объём памяти, необходимый для содержаний всей таблицы ключей: 23 \* 40 = 920 байт (это всего лишь 13% от памяти для всей таблицы).

**Контрольные вопросы**

**Как выделяется память под вариантную часть записи?**

В языке си вариантная часть может быть реализована с помощью объединений. Объединения могут иметь несколько полей, однако количество памяти, выделяемое под объединение, равно размеру максимального по длине полю.

**Что будет, если в вариантную часть ввести данные, несоответствующие** **описанным?**

Транслятор не контролирует тип данных, который заносится в вариантную часть, поэтому данные будут записаны, что повлечёт ошибки в дальнейшей работе программы.

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

Исключительно программист.

**Что представляет собой таблица ключей, зачем она нужна?**

Таблица ключей - это дополнительная таблица, содержащая индекс элемента в исходной таблице и значения выбранного ключевого поля (по которому, например, будет производиться сортировка всей таблицы). Таблица ключей экономит время обработки основной таблицы.

**В каких случаях эффективнее обрабатывать данные в самой таблице, а когда – использовать таблицу ключей?**

Если исходная таблица небольшая и содержит малое количество записей, то эффективнее сортировать её сразу всю. Однако, если же массив данных большой, то эффективнее использовать таблицу ключей.

**Какие способы сортировки предпочтительнее для обработки таблиц и**

**почему?**  
 Предпочтительнее те способы сортировки, которые не переставляют уже отсортированные элементы, так как перестановка элементов таблицы занимает много времени. Например, сортировка пузырька с флагом или метод сортировки вставками.

**Вывод**

В процессе выполнения данной лабораторной работы я ознакомилась с принципами реализации записей с вариантами и способами обработки таблиц.

В заключение хочу сказать, что периодически могут возникнуть такие ситуации, когда программисту необходимо будет выбрать наиболее эффективный способ обработки данных вариантных записей.