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 №2 «ЗАПИСИ С ВАРИАНТАМИ. ОБРАБОТКА ТАБЛИЦ»

Студент Хамзина Регина Ренатовна
фамилия, имя, отчество

Студент, группа Хамзина Р.Р., ИУ7-33Б

2020 г.

Описание условия задачи

Создать таблицу, содержащую не менее 40-ка записей (тип – запись с вариантами

(объединениями)). Упорядочить данные в ней по возрастанию ключей, двумя алгоритмами сортировки, где ключ – номер абонента, используя: а) саму таблицу, б) массив ключей. (Возможность добавления и удаления записей в ручном режиме обязательна). Осуществить поиск информации по варианту.
Для 19-ого варианта:
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.

Техническое задание

Входные данные

Файл: текстовый файл формата txt, который содержит данные для записи в исходную таблицу. Каждая запись таблицы находится на новой строке файла. Поля в записи разделены символом «;».
Поля записи:
1. Фамилия
2. Имя
3. Номер телефона
4. Адрес
5. Статус абонента:
    1. Личный:
        1. День рождения
        2. Месяц рождения
        3. Год рождения
    2. Служебный:
        1. Должность
        2. Организация

Целое число: номер команды, целое число в интервале от 0 до 12

Дополнительный ввод: строка, представляющая собой данные абонента, разделенные «;» или строка, представляющая собой номер телефона абонента в зависимости от пункта меню

Выходные данные

Текущее состояние таблицы, текущее состояние таблицы ключей, сравнение сортировок, список, состоящий из элементов таблицы, удовлетворяющих условию.

Функция программы

Программа вводит, выводит и обрабатывает таблицу записей-абонентов при помощи меню.

Обращение к программе

Программа запускается из терминала командой «./app.exe» в директории с программой.

Возможные аварийные ситуации и ошибки пользователя

1. Неверный ввод пункта меню
   Вход: не число, число, не входящее в диапозон — от 0 до 12 включительно
   Выход: сообщение «Необходимо вводить только числа, равные пунктам меню, от 0 до 12.»

2. Неверный ввод имени файла
   Вход: строка с отсутствующим файлом
   Выход: сообщение «Такого файла не существует.»

3. Пустой файл
   Вход: строка с именем пустого файла
   Выход: сообщение «Файл пустой.»

4. Число записей в файле больше допустимого
   Вход: строка с именем файла, в котором число записей больше допустимого
   Выход: сообщение «Число записей в файле больше допустимого.»

5. Неверный ввод фамилии
   Вход: длина фамилии больше 10 символов
   Выход: сообщение «Длина фамилии должна быть меньше 11 символов.»

6. Неверный ввод имени
   Вход: длина имени больше 10 символов
   Выход: сообщение «Длина имени должна быть меньше 11 символов.»

7. Неверный ввод номера телефона
   Вход: длина номера телефона больше 7 символов
   Выход: сообщение «Длина номера должна быть меньше 8 символов.»

8. Неверный ввод номера телефона
Вход: номер телефона состоит не из цифр
Выход: сообщение «Номер абонента должен состоять только из цифр.»

9. Неверный ввод адреса
Вход: длина адреса больше 20 символов
Выход: сообщение «Длина адреса должна быть меньше 21 символа.»

10. Неверный ввод статуса абонента
Вход: не «0» и не «1»
Выход: сообщение «Должен быть указан тип статуса - 0 или 1.»

11. Неверный ввод дня рождения
Вход: введено не число, отрицательное число или больше 31
Выход: сообщение «День рождения должен быть числом от 0 до 31 включительно.»

12. Неверный ввод месяца рождения
Вход: длина месяца больше 9 символов
Выход: сообщение «Длина месяца должна быть меньше 10 символов.»

13. Неверный ввод года рождения
Вход: не число или отрицательное число
Выход: сообщение «Год должен быть целым числом, большим 0.»

14. Неверный ввод должности
Вход: длина должности больше 20 символов
Выход: сообщение «Длина должности должна быть меньше 21 символа.»

15. Неверный ввод организации
Вход: длина организации больше 20 символов
Выход: сообщение «Длина организации должна быть меньше 21 символа.»

Внутренняя структура данных

Так как **ввод**, **вывод** и **обработка** записей возможна в двух вариантах: при помощи таблицы записей и при помощи таблицы ключей, используется следующая структура данных.
Структурный тип *table_t*, поля которого — **таблица записей** - массив элементов структурного типа *human,* **таблица ключей -** массив элементов структурного типа *keys,* и размер этих таблиц. *COUNT_SUBSCRIBER* — максимальное допустимое число записей, равное 1000:

typedef struct
{
subscriber human[COUNT_SUBSCRIBER];
keys key[COUNT_SUBSCRIBER];
int table_size;
} table_t;

Элемент структурного типа *keys* — ключ записи - имеет поля *index* — индекс текущей записи в таблице и *number* — номер абонента. *MAX_COUNT_DIGITS* — максимальное число цифр в номере, равное 7.

typedef struct
{
int index;
char number[MAX_COUNT_DIGITS + 1];
} keys;

Элемент структурного типа *subscriber* — абонент - имеет поля *surname* — фамилия абонента, *name* - имя, *number* - номер, *address* - адрес, поле структурного типа *status_t status_type* — статус абонента и *variable* смесь — поля зависят от статуса. *MAX_LEN_SURNAME* — максимальная длина фамилии, равная 10, *MAX_LEN_NAME* — максимальная длина имени, равная 10, *MAX_COUNT_DIGITS* — максимальное число цифр в номере, равное 7 *MAX_LEN_ADDRESS* — максимаьная длина адреса, равная 20.

typedef struct
{
char surname[MAX_LEN_SURNAME + 1];
char name[MAX_LEN_NAME + 1];
char number[MAX_COUNT_DIGITS + 1];
char address[MAX_LEN_ADDRESS + 1];
status_t status_type;
variable human_status;
} subscriber;

Перечисляемый тип status_t используется для описания типа статуса: *personal* — личный статус, *service* — служебный статус.

typedef enum status
{
personal,
service
} status_t;

Смесь *variable* представляет собой вариантное поле — статус абонента. Структурная переменная типа *personal_t* — личный статус, *service_t* — служебный.

```
typedef union
{
personal_t personal_status;
service_t service_status;
} variable;
```

Личный статус состоит из трех полей — *day* — день рождения, *month* — месяц рождения и *year* — год рождения. *MAX_LEN_MONTH* — маскимальная длина месяца рождения, равная 9.

```
typedef struct
{
int day;
char month[MAX_LEN_MONTH + 1];
int year;
} personal_t;
```

Служенбый статус состоит из двух полей — *position* — должности и *organization* — организации. *MAX_LEN_POSITION* — маскимальная длина должности, равная 20, *MAX_LEN_ORGANIZATION* - маскимальная длина организации, равная 20.

```
typedef struct
{
char position[MAX_LEN_POSITION + 1];
char orgaization[MAX_LEN_ORGANIZATION + 1];
} service_t;
```

Алгоритм

Диалог с пользователем осуществляется при помощи меню, в конце которого предлагается выбрать один из пунктов. Выбранный пункт соответствует определенному действию. Выполняется до тех пор, пока пользователь не выберет пункт выхода.

Функции программы

1. Функция ввода команды

*int input_command(int *command);*

2. Функция ввода данных из файла

*int read_data(table_t *table, char *filename);*

3.Функция печати таблицы

void print_table(table_t table);

4.Функция печати таблицы с помощью ключей

void print_table_by_key(table_t table);

5.Функция печати таблицы ключей

void print_key_table(table_t table);

6. Функция поиска друзей, которых необходимо поздравить с днем рождения

*int print_bithday_friends(table_t *table);*

7. Добавление абонента
*int add_subscriber(table_t *table, char *record);*

8.Удаление абонента
*int delete_subscribers(table_t *table, char *record);*

9. Быстрая сортировка
*void sort_table_qsort(table_t *table, int type_sort, int start, int end);*

10. Сортировка пузырьком
*void sort_table_bubble(table_t *table, int type_sort);*

11. Печать результатов сортировки
*void print_sorts_result(table_t *table, char *filename);*

Тесты

№	Случай	Ввод	Вывод
1	Неверный ввод команды	a	Необходимо вводить только числа, равные пунктам меню, от 0 до 12.
2	Неверный ввод команды	16	Необходимо вводить только числа, равные пунктам меню, от 0 до 12.
3	Неверное имя файла	no.txt	Такого файла не существует.
4	Пустой файл	empty.txt	Файл пустой.
5	Число записей в файле больше допустимого	Файл с числом записей больше 1000	Число записей в файле больше 1000.
6	Число записей в таблице переполнено	При добавлении записи, записей в таблице стало больше 1000	Таблица заполнена полностью.
7	Неверный ввод фамилии	Длина фамилии больше 10	Длина фамилии должна быть меньше 11 символов.
8	Неверный ввод имени	Длина имени больше 10	Длина имени должна быть меньше 11 символов.
9	Неверный ввод номера телефона	Номер телефона состоит более, чем из 7 цифр	Длина номера должна быть меньше 8

			символов.
10	Неверный ввод номера телефона	Номер телефона состоит не из цифр	Номер абонента должен состоять только из цифр.
11	Неверный ввод адреса	Длина адреса больше 20	Длина адреса должна быть меньше 21 символа.
12	Неверный ввод статуса абонента	Не 0 и не 1	Должен быть указан тип статуса - 0 или 1.
13	Неверный ввод дня рождения	Не число или отрицательное число или 0 или больше 31	День рождения должен быть числом от 0 до 31 включительно.
14	Неверный ввод месяца рождения	Длина месяца больше 9	Длина месяца должна быть меньше 10 символов.
15	Неверный ввод года рождения	Не число или отрицательное число или 0	Год должен быть целым числом, большим 0.
16	Неверный ввод должности	Длина должности больше 20	Длина должности должна быть меньше 21 символа.
17	Неверный ввод организации	Длина организации больше 20	Длина организации должна быть меньше 21 символа.
18	Корректный ввод записей из файла	Файл с корректными записями	Данные записаны в таблицу.
19	Добавление верной записи	Верная запись	Абонент добавлен.
20	Удаление записи по номеру, которая есть	Существующий номер	Абоненты удалены.
21	Удаление записи	Несуществующий	Абоненты с

	по номеру, которой нет	номер	данным номером не найдены.
22	Поиск по условию, записи найдены	Дата, удовлетворяющая условию	Список друзей, удовлетворяющих условию
23	Поиск по условию, записи не найдены	Дата, не удовлетворяющая условию	Таких друзей нет.
24	Любая сортировка	Непустая таблица	Отсортированная таблица
25	Сравнение сортировок	Непустая таблица	Результаты сравнения сортировоки
26	Вывод любой непустой таблицы	Непустая таблица	Таблица с записями
27	Вывод любой пустой таблицы	Пустая таблица	Необходимо ввести данные из файла в таблицу или добавить абонента в таблицу.
28	Выбор пункта меню «0»	0	Успешное завершение работы программы

ОЦЕНКА ЭФФЕКТИВНОСТИ

Время сортировки в секундах:

Число записей	Полная таблица		Таблица ключей	
	Сортировка пузырьком	Быстрая сортировка	Сортировка пузырьком	Быстрая сортировка
30	0.039345	0.014101	0.024399	0.002822
150	0.727786	0.105607	0.522634	0.019556
300	2.852248	0.352024	2.095232	0.067696

Объем занимаемой памяти в байтах:

Число записей	Полная таблица	Таблица ключей
30	3360	360
150	16800	1800
300	33600	3600

Таблица соотношения памяти и времени:

Число записей	% памяти, занимаемой таблицей ключей от полной таблицы	T полной/ T ключей пузырьком	T полной/ T ключей быстрой сортировкой
30	~11%	~1.6 раз	~5 раз
150	~11%	~1.4 раз	~5.4 раз
300	~11%	~1.3 раз	~5.2 раз

ОТВЕТЫ НА КОНТРОЛЬНЫЕ ВОПРОСЫ

1.Как выделяется память под вариантную часть записи?

Выделяется общая для всех полей вариантной части записи память. Ее размер равен максимальному по длине полю вариантной части.

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

Поведение программы в этом случае не определенно, так как в вариантной части при компиляции тип данных не проверяется и невозможно считать данные корректно.

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

За правильностью выполнения операций должен следить программист.

4.Что представляет собой таблица ключей, зачем она нужна?

Таблица ключей — структура, поля которой массивы индексов и поля. Таблица ключей необходима для ускорения обработки таблиц записей.

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

В случае, когда исходная таблица содержит небольшое количество полей, то эффективнее обрабатывать данные в самой таблице. Иначе - использовать таблицу ключей.

6.Какие способы сортировки предпочтительнее для обработки таблиц и почему?

При сортировке по таблице ключей эффективнее использовать сортировки с наименьшей сложностью работы В случае сортировки исходной таблицы, необходимо использовать алгоритмы, требующие наименьшее количество операций перестановки.

Выводы

В лабораторной работе реализована обработка :
1) вариантных записей. Вариантная запись экономит память, так как выделятся под одно самое длинное поле. Однако правильность данных этой структуры должна быть проверена программистом, поскольку компилятор не сможет отследить ошибку.
2) таблицы записей при помощи таблицы ключей.
В лабораторной работе производилась сортировка пузырьком и быстрая сортировка по строковому полю.
Для сортировки пузырьком по строковому полю использование таблицы ключей неоправданно, так как таблица ключей занимает дополнительную память, не сильно сокращая время работы. При этом с увеличением числа записи таблицы, сокращение времени работы уменьшалось.
Для быстрой сортировки по строковому полю использование таблицы ключей сокращает время обработки в 5 раз, занимая дополнительную память. С увеличением числа записей в таблице сокращение времени работы увеличилось незначительно.
Таким образом, эффективность использования таблицы ключей для обработки таблицы записей зависит от конкретной ситуации. Удобно использовать этот способ обработки в случаях, когда поля таблицы занимают небольшую память и число записей в таблице достаточно большое.