



Министерство образования и науки 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**

**По дисциплине «Типы и структуре данных»**

Название Обработка таблиц

Студент Назирова Илхомджон Вохиджон  
*фамилия, имя, отчество*

Группа ИУ7-35Б

Тип лабораторной работы Учебная

Название  
предприятия

---

Студент \_\_\_\_\_ И.В. \_\_\_\_\_ Назирова И.В.  
*подпись, дата* *фамилия, и.о.*

Преподаватель \_\_\_\_\_ \_\_\_\_\_ Никульшина Т.А.  
*подпись, дата* *фамилия, и.о.*

## Цель работы.

Приобрести навыки работы с типом данных «запись» (структура), содержащим вариантную часть (объединение, смесь), и с данными, хранящимися в таблицах, произвести сравнительный анализ реализации алгоритмов сортировки и поиска информации в таблицах, при использовании записей с большим числом полей, и тех же алгоритмов, при использовании таблицы ключей; оценить эффективность программы по времени и по используемому объему памяти; оценить эффективность использования различных алгоритмов сортировок.

## Условие 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 – любое невариантное поле (по выбору программиста), используя:

- а) саму таблицу,
- б) массив ключей.

(Возможность добавления и удаления

записей в ручном режиме обязательна). Осуществить

поиск информации по варианту.

Ввести список стран, в которых можно отдохнуть, содержащий название страны, количество жителей, столицу, материк, нужна ли прививка или ПЦР, основной вид туризма

(экскурсионный - количество объектов, основной вид (природа, история, искусство); пляжный – основной сезон, температура воздуха и воды, время полета до страны; спортивный – вид спорта (горные лыжи, серфинг, восхождения), минимальная

стоимость отдыха.). Вывести список стран на выбранном материке, где можно заняться указанным видом спорта.

## Требования к программе:

1. Задача решается на языке программирования "с".
2. При некорректном вводе пользователя выводится сообщение об ошибке и просьба ввести заново.
3. Сортировки должны быть различны по эффективности (для сравнения)
4. Указание действий, производимых программой
5. Наличие пояснений при выводе результата
6. Возможность добавления записей в конец таблицы и удаления записи по значению указанного поля.

7. Вывод результатов сравнения эффективности работы программы при обработке данных в исходной таблице и в таблице ключей;

# Техническое задание.

## Входные данные.

1. Цифра, которая указывает на пункт меню, который необходимо выполнить.
2. Информационные данные:
  - i. Числа, определяющие какой-то признак (например кол-во людей в стране)
  - ii. Строки, определяющие какой-то признак (например название страны)
3. Файл, в котором лежит таблица с странами

## Выходные данные.

1. Таблица
2. Таблица ключей
3. Таблица, которая выводится по ключам
4. Результаты поиска
5. Время сортировки

## Описание задачи, реализуемой программой.

Задача программы состоит из нескольких частей:

1. Работа с данными: добавление в таблицу (из файла или со стандартного потока ввода) и удаление из таблицы;
2. Форматный вывод основной таблицы, таблицы ключей и условной таблицы (минимальная стоимость отдыха);
3. Сортировка данных в таблице двумя способами: напрямую и через таблицу ключей;
4. Сравнение эффективности различных методов сортировки таблицы;

## Способ обращения к программе.

Программа реализована как консольное приложение с расширением .exe. Запускается либо из графического проводника имеющейся ОС, либо в консоли с помощью команды запуска исполняемого файла.

## Аварийные ситуации и ошибки пользователя.

Ошибка	Сообщение
Ошибка ввода опции меню	<pre>"Sorry which variant you choose is not exists"</pre>

Попытка провести операции с пустой таблицей (кроме добавления записей)	"Empty File!!!"
Ошибка открытия файла	"Error opening file!"
Пустой файл	"File is empty!"
Переполнение таблицы	"Table overflow!"
Неправильный ввод данных	"Invalid data please enter correct data!!!"
Неправильные данные в файле	"Error data please check the data and try again!!!"

# Алгоритм.

`int main(int argc, char **argv):`

- Создание основной таблицы и таблицы ключей, вывод основного меню;

`int read_csv(FILE *f):`

- Проверка файла на ошибки: наличие, пустота, неизвестного рода ошибки;
- Считывание структур и запись данных в структуру;

`void print_table (st_travel *travel_data, st_key *key_data, int st_len):`

- Печать главной таблицы;

`void print_key_table (st_key *key, int st_len):`

- Печать таблицы ключей;

`int st_search(st_travel *travel_data, st_travel *sres_travel_data, int st_len, int *rs_len)`

- Поиск сказок для детей указанного возраста, вывод таблицы сказок для детей указанного возраста, вывод сообщения об отсутствии удовлетворительных записей или вывод сообщения о пустой таблице;

`void st_bsort(st_travel *travel_data, st_key *key_data, int st_len);:`

- Сортировка главной таблицы «пузырьком»;

`void st_selection_sort(st_travel *travel_data, st_key *key_data, int st_len);:`

- «Выборкой» сортировка главной таблицы;

`void st_bsort_by_key (st_key *key_data, int st_len);:`

- Сортировка главной таблицы ключей «пузырьком»;

`void st_selection_sort(st_key *key_data, int st_len);:`

- «Выборкой» сортировка таблицы ключей;

# Внутренние структуры данных.

Структура с данными о достопримечательности

```
typedef struct
{
    char essential_type_of_tourism[LEN_OF_TOURISM]; // Основной вид туризма и
стране
    char feature_of_turism[LEN_OF_FEATURE]; // параметры этого туризма
    int min_budget;

} st_tourism;
```

Структура с ключевыми данными о спектаклях:

```
typedef struct
{
    int id; // Номер записи в основной таблице;
    int key; // Минимальная стоимость отдыха;
} st_key;
```

Структура с данными о путешествии:

```
typedef struct
{
    char country[LEN_OF_COUNTRY]; // Название страны
    char capital[LEN_OF_CAPITAL]; // Название столицы
    char materic[LEN_OF_MATERIC]; // Название материка
    long int demography; // Кол людей живущих в стране
    int time_of_way; // Сколько минут занимает время полета до
страны
    short is_requierd_vacinie; // Нужен ли вакцина для отдыха в этой
стране
    short is_required_pts_test; // Нужен ли тест на коронавирус
} st_travel;
```

Объединение структуры достопримечательности и структуры путешествии:

```
typedef union
{
    st_travel travel;
    st_tourism tourism;
```

```
} travel_union;
```

## Тесты.

Тестируемая часть	Вывод
Выбор опции главного меню	
Попытка добавить записи в переполненную таблицу	TABLE OVER FLOW!!!
Ввод имени несуществующего файла	Error opening file!
Ввод имени пустого файла	Empty File!!!
Проведение операций с пустой таблицей	Empty Table!!!
Ввод индекса удаляемой записи	Data successfully deleted
Выход за диапазон допустимых значений	OVER FLOW!!!
Пустой ввод	Invalid data please enter correct data!!!
Некорректный символ	Invalid data please enter correct data!!!
Некорректный ввод для поиска	Invalid data please enter correct data!!!
Отсутствие записей, удовлетворяющих условию	Not such elements in file



# Замеры времени

Время сортировки |SELECTION\_SORT| QSORT|

Количество элементов 100:

```
Сравнение сортировки таблицы Selection sort и QSORT
|-----|-----|-----|
|      TICK |   SIZE | S_SORT |
|-----|-----|-----|
|    224283 |     608 |     89 |
|-----|-----|-----|

|-----|-----|-----|
|      TICK |   SIZE |  QSORT |
|-----|-----|-----|
|    127429 |     608 |     55 |
|-----|-----|-----|
```

```
Сравнение Selection sort и qsort для сортировки ключей.
|-----|-----|-----|
|      TICK |   SIZE | S_SORT |
|-----|-----|-----|
|    79350 |      8 |     35 |
|-----|-----|-----|

|-----|-----|-----|
|      TICK |   SIZE |  QSORT |
|-----|-----|-----|
|    18388 |      8 |      8 |
|-----|-----|-----|
```

Количество элементов 800:

Сравнение сортировки таблицы Selection sort и QSORT

TICK	SIZE	S_SORT
4313114	608	1871

TICK	SIZE	QSORT
467625	608	204

Сравнение Selection sort и qsort для сортировки ключей.

TICK	SIZE	S_SORT
3174563	8	1383

TICK	SIZE	QSORT
83582	8	37

**Количество элементов 1176:**

Сравнение сортировки таблицы Selection sort и QSORT

TICK	SIZE	S_SORT
10037320	608	4366

TICK	SIZE	QSORT
558452	608	243

Сравнение Selection sort и qsort для сортировки ключей.

TICK	SIZE	S_SORT
5880003	8	2562

TICK	SIZE	QSORT
133425	8	58

## Сравнение в процентах

Эффективность сортировки таблицы по ключу с помощью алгоритма выборкой составляет 38-42%.

Эффективность сортировки таблицы по ключу с помощью алгоритма QSORT составляет 75-83%.

Сортировка таблицы с помощью ключа дает нам преимущество в скорости но за это преимущество мы платим памятью . В моем случае один ключа занимает 8 байт. И количество выделяемой памяти под ключ зависит от количество элементов в таблице  $8 * N$ (таблица ключей занимает 1.5% от всего таблицы в моем случае). Здесь N количество элементов в таблице

# Ответы на вопросы.

## 1. Как выделяется память под вариантную часть записи?

В языке Си, вариативная часть структуры реализована с помощью union. Память выделяется в одном "куске" памяти, имеющий размер, который способен вместить наибольшее поле из указанных.

## 2. Что будет, если в вариантную часть ввести данные, не соответствующие описанным?

Невозможно корректно прочитать данные.

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

Результат будет системно-зависимым и трудно предсказуемым. Возможно, произойдет приведение типов. Кто должен следить за правильностью выполнения операций с вариантной частью записи? Ответственность за правильность проведения операций целиком и полностью лежит на программисте. "Следить и помнить, какие именно данные были помещены в объединение, - это забота программиста" — Брайан Керниган, Деннис Ритчи. Язык программирования Си.

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

При больших размерах таблиц поиск данных и их сортировка может потребовать больших затрат времени. В этом случае можно уменьшить время обработки за счет создания дополнительного массива – таблицы ключей, содержащей индекс элемента в исходной таблице и выбранный ключ.

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

Обрабатывать данные в самой таблице эффективнее использовать когда время обработки не так важно, как задействованная память. А использование таблицы ключей, наоборот, эффективно когда нужно быстрое время обработки и не так важна дополнительная задействованная память. Также, использование таблицы неэффективно, когда сама таблица состоит из маленького количества полей, например, таблица, имеющая два поля: "Ученик" и "Оценка". В таком случае, таблица ключей будет лишь занимать дополнительное место в памяти и не даст никакой выгоды во времени.

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

Для таблиц из большого количества записей предпочтительно использовать стандартные и устойчивые способы сортировки, со средним временем обработки  $O(n \cdot \log n)$ , такие как QuickSort, MergeSort и т.д. Если же в таблице не так много записей, то предпочтительнее использовать простые алгоритмы сортировки, например, сортировку пузырьком.

## **Выводы по проделанной работе**

Чем больше размер самой таблицы, тем более эффективнее становится сортировка массива ключей, но даже на маленьких размерах таблицы, сортировка массива ключей происходит быстрее, чем сортировка самой таблицы. Однако, для хранения массива ключей нужна дополнительная память. В моем случае понадобилось не так много дополнительной памяти под массив ключей, но, если бы я в качестве поля выбрал бы не площадь квартиры (тип int, 4б), а, например, строку, длиной хотя бы 10 символов, то затраты на память возросли. Стоит отметить, что использование массива ключей неэффективно при небольших размерах самой таблицы. В таком случае эффективнее просто отсортировать таблицу, так как разница во времени будет несущественна, а затраты на память сократятся.