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 №1 ПО ДИСЦИПЛИНЕ: ТИПЫ И СТРУКТУРЫ ДАННЫХ

“ЗАПИСИ С ВАРИАНТАМИ. ОБРАБОТКА ТАБЛИЦ”

Студент **Доколин Георгий Александрович**

Группа **ИУ7-32Б**

Вариант: **5**

Название предприятия **НУК ИУ МГТУ им. Н. Э. Баумана**

Оценка _____

Преподаватель _____

2024 г.

Оглавление

Описание условия задачи

Создать таблицу, содержащую не менее 40 записей с вариантной частью. Произвести поиск информации по вариантному полю. Упорядочить таблицу, по возрастанию ключей (где ключ – любое невариантное поле по выбору программиста), используя: а) исходную таблицу; б) массив ключей, используя 2 разных алгоритма сортировки (простой, ускоренный). Оценить эффективность этих алгоритмов (по времени и по используемому объему памяти) при различной реализации программы, то есть, в случаях а) и б). Обосновать выбор алгоритмов сортировки. Оценка эффективности должна быть относительной (в %).

Дополнительные условия по варианту номер 5:

Ввести репертуар театров, содержащий: название театра, название

спектакля, диапазон цен на билеты,

тип спектакля:

1. Пьеса
2. Драма
3. Комедия
4. Сказка
    a. Возраст: 3+, 10+, 16+
5. Музыкальный:
    a. Композитор
    b. Страна
    c. Вид: балет, опера, мюзикл
    d. Возраст: 3+, 10+, 16+
    e. Продолжительность

Вывести список всех балетов для детей указанного возраста, продолжительностью меньше указанной.

Описание технического задания

Входные данные:

- Выбор действия в пункте меню выполнен в виде ввод целого числа от 1 до 9 включительно:
  - 1 - Добавить элементы
  - 2 - Удалить элемент
  - 3 - Завершить программу
  - 4 - Вывести искомые значения
  - 5 - Вывести всё
  - 6 - Считать базу данных в массив
  - 7 - Отсортировать массив с структурами по начальной цене
  - 8 - Отсортировать массив с ключами по цене
  - 9 - Сделать замеры
- Входной файл может включать в себя от 0 до 1000 элементов, состоящиъ из полей:
  - Название театра – строка, длиной от 1 до 25 символов
  - Название спектакля – строка, длиной от 1 до 25 символов
  - Минимальная цена за билет – вещественное, не отрицательное число
  - Максимальная цена за билет – вещественное, не отрицательное число
  - Жанр спектакля – целое не отрицательное число, обозначающее выбор пользователя (поскольку пользователю предоставлен числовой выбор, эквивалентный строковому значению)
  - Имя композитора – строка, длиной от 1 до 25 символов
  - Страна происхождения – строка, длиной от 1 до 25 символов
  - Вид музыкального шоу – целое не отрицательное число, обозначающее выбор пользователя (поскольку пользователю предоставлен числовой выбор, эквивалентный строковому значению)

    - Минимальный возраст - целое не отрицательное число, обозначающее выбор пользователя (поскольку пользователю предоставлен числовой выбор, эквивалентный строковому значению)
    - Продолжительность – целое не отрицательное число.
- При удалении программа запршивает у пользователя название спектакля (массив символов, длиной от 1 до 25 символов), а так же количество первых записей, которые необходимо удалить.
- При поиске элементов, необходимо будет ввести возраст (целое неотрициательное число, которое будет эквивалентно выбору из пунка меню), а также максимальну продолжительность спектакля.

Выходные данные:

- Таблица записей в исходном виде
- Отсортированная по ценам таблица записей
- Таблица ключей в исходно виде
- Отсортированная по ценам таблица ключей
- Таблица записей, которые подошли по по параметрам поиска
- Таблица с результатами замеров сортировок

Реализуемые программой задачи:

- Добавить элементы в конец
- Удалить элементы по названию шоу
- Завершить программу
- Вывести искомые значения, задав параметры возраста, а также продолжительности
- Вывод таблицы записей, а также таблицы ключей
- Считать базу данных в массив – обновить массив, считав данные из файла заново в массив
- Отсортировать массив с структурами по начальной цене
- Отсортировать массив с ключами по цене
- Сделать замеры сортировок и вывести результаты

Способ обращения к прогрмме

Обращение к программе выполняется через командную строку посредством ввода команды ./app.exe [имя файла]

Возможные ошибки в программе

- Ошибка открытия файла – Такого файла не существует или иная проблема с открытием
- Ошибка длины файл – файл пуст
- Ошибка количества аргументов
- Ошибка количества структур в файле
- Ошибка при вводе названия театра
- Ошибка при вводе названия спектакля
- Ошибка при вводе минимальной стоимости
- Ошибка при вводе максимальной стоимости
- Ошибка правильности определения цен (минимальная цена, больше максимальной)
- Ошибка при вводе жанра спектакля
- Ошибка при вводе имени композитора
- Ошибка при вводе названия страны
- Ошибка при вводе типа музыкального спектакля
- Ошибка при вводе минимального возраста
- Ошибка при вводе продолжительности
- Ошибка при ввыборе пункта меню
- Ошибка удаления элемента
- Ошибка при замерах, поскольку элементов менее 1000

Аварийное завершение программы

Аварийное завершение программы возможно только в случае некорректных данных в файле.

Описание структур

Структура, хранящая в себе все данные файла

```
typedef struct
{
   char name_theater[LEN_NAME_THEATER];
   char name_show[LEN_NAME_SHOW];
   double start_price;
   double finish_price;
   int type_of_show;
   union{
      unsigned int age_start;
      struct
      {
         char name_composer[LEN_NAME_COMPOSER];
         char country[LEN_COUNTRY];
         int type_of_musical;
         int age_start;
         int duration;
      } add_character;
   } character;
} show_info;
```

LEN_NAME_THEATER – Константа, хранящая в себе длину массива символов, в который записывается название театра. LEN_NAME_THEATER = 25.

LEN_NAME_SHOW – Константа, хранящая в себе длину массива символов, в который записывается название спектакля. LEN_NAME_SHOW = 25.

LEN_NAME_COMPOSER – Константа, хранящая в себе длину массива символов, в который записывается имя композитора. LEN_NAME_COMPOSER = 25.

LEN_COUNTRY – Константа, хранящая в себе длину массива символов, в который записывается название страны. LEN_COUNTRY = 25.

Переменные:

- char name_theater - название театра – массив символов, длиной от 1 до 25 символов
- char name_show - массив символов, длиной от 1 до 25 символов
- double start_price - минимальная цена за билет, представленная в виде вещественного, не отрицательного числа
- double finish_price - максимальная цена за билет, представленная в виде вещественного, не отрицательного числа
- int type_of_show - жанр спектакля, представлен в виде целого не отрицательного числа (поскольку пользователю предоставлен числовой выбор, эквивалентный строковому значению)
- char name_composer[LEN_NAME_COMPOSER] - Имя композитора – массив символов, длиной от 1 до 25 символов
- char country[LEN_COUNTRY] - страна происхождения – массив символов, длиной от 1 до 25 символов
- int type_of_musical - вид музыкального шоу, представлен в виде целого не отрицательного числа (поскольку пользователю предоставлен числовой выбор, эквивалентный строковому значению)
- int age_start - минимальный возраст, представленный в виде, целое не отрицательнго числа (поскольку пользователю предоставлен числовой выбор, эквивалентный строковому значению)
- int duration – продолжительность, представленная в виде целого не отрицательного числа.

Структура, хранящая в себе значение индекса в таблице, а также минимальную цену билета по этому индексу.

```
typedef struct
{
    int index;
    double start_price;
} key_index;
```

index – целочисленная переменная, хранящая индекс объекта в таблице

double start_price - минимальная цена за билет, представленная в виде вещественного, не отрицательного числа

Обоснование целесообразности применения записи с вариантной частью

Данная запись имеет смысл, поскольку она позволяет экономить место, используемое программой для работы. В моей структуре видно это преимущество, которое заключается в выделении максимального блока памяти, в котором мы можем хранить или одно значение, или несколько, но не оба сразу, поскольку это и не возможно программой, данное действие помогает экономить место для отдельной переменной, которая не всегда нужна.

Основной алгоритм программы

1. Открывается файл
2. Проверяется, длина файла, если она равна 0, то пользователю предлагается ввести данные. В случае неверного ввода, он будет предлагаться до тех пор, пока корректно не будет введна хотя бы одна запись.
3. Программа считывает данные из файла в массив, проверяя каждый элемент на корректность (строки – на длину, а числа на диапазон допустимых значений, а также на тип вводимого числа), если есть неправильная запись, то программа аварийно завершается с сообщением, при вводе какого элемента структура была совершена ошибка.
4. После успешного завершения предыдущих пунктов пользователю открывается меню. С этого момента программа не может завершиться аварийно, поскольку при вводе некорректных данных в любом месте программы, пользователь будет возвращен в главное меню. Программу можно завершить при помощи выбора соответствующего пункта меню.
5. Функции меню
    1. Добавить элементы

       Данная функция добавляет элемент в конец файла, проверяя количество уже введенных данных, а также все данные вводимые пользователем перед записью в файл. После записи элемента в файл, пользователю предлагается продолжить ввод, в случае согласия, повторяются предыдущие шаги, а в случае отказа, происходит считывание всего файла заново в массив и выход в главное меню.
    2. Удалить элемент

       Данная функция запрашивает у пользователя назавние спектакля, а также количество записей, которые он хочет удалить с этим названием.
    3. Завершить программу

       Данная функция завершает программу.
    4. Вывести искомые значения

       Данная функция запрашивает у пользователя минимальный возраст, максимальную продолжительность спектакля, а также берет для

сравнения данные, указанные в задании, тоесть жанр должен быть музыкальным, а тип музыкального спектакля – балет. Программа сравнивает все записи с искомыми и в случае совпадения выводит их в виде таблицы.

5. Вывести всё

   Функция выводит таблицу с данными, а также таблицу ключей.

6. Считать базу данных в массив

   Функция считывает данные из файла в массив заново.

7. Отсортировать массив с структурами по начальной цене

   Функция сортирует таблицу с данными по возрастанию минимальной цены за билет и выводит результат

8. Отсортировать массив с ключами по цене

   Функция сортирует таблицу ключей по возрастанию минимальной цены и выводит таблицу данный при помощи индексов, а также выводит таблицу ключей в отсортированном виде.

9. Сделать замеры

   Функция совершает замеры сортировок массивов разных длин, при помощи разных методов: сортировка пузырьком и быстрая сортировка. Также проводятся замеры сортировки массива ключей при помощи тех же видо сортировок. Результаты выводтся в виде таблицы.

Реализованные функции

int add_to_file(char **argv, size_t *cnt_data_in_file)

- **Назначение:** Добавление элементов в конец файла.
- **Входные данные:** Аргументы коммандной строки, адрес на количество элементов массива.
- **Выходные данные:** Код ошибки или код об успешном завершении работы функции.

int input_in_file_one_elem(FILE *file_out, int cnt, size_t len_arr)

- **Назначение:** Считвание, проверка и запись одного элемента.
- **Входные данные:** Файла, длина массива
- **Выходные данные:** Код ошибки или код об успешном завершении работы функции.

int read_to_arr(char **argv, show_info arr_show_info[], size_t *len_arr_show_info)

- **Назначение:** Считвание данных из файла в массив.
- **Входные данные:** Аргументы коммандной строки, массив структур, хранящих все записи, адрес на длину этого массива.
- **Выходные данные:** Код ошибки или код об успешном завершении работоы функции.

int read_item(FILE *file_in, show_info arr_show_info[], size_t cnt_elems_arr)

- **Назначение:** Считвание одной структуры из файла в массив.
- **Входные данные:** Файла, массив, хранящий все записи, длина массива.
- **Выходные данные:** Код ошибки или код об успешном завершении работы функции.

void create_arr_keys(show_info arr_show_info[], key_index arr_key_index[], size_t len_arr_show_info)

- **Назначение:** Создание массива структур, хранящих индексы и ключи записей.
- **Входные данные:** Массив, хранящий все записи, массив, хранящий структуры с ключами и индексами, длина массива с записями.

- **Выходные данные:** Отсутствуют.

void work_with_errors(int rc)

- **Назначение:** Обработка ошибок.
- **Входные данные:** Код ошибки.
- **Выходные данные:** Отсутствуют

void clean_enter()

- **Назначение:** Считывание символа переноса строки после функции **fscanf**.
- **Входные данные:** Отсутствуют.
- **Выходные данные:** Отсутствуют.

int input_data_from_array(char **argv, show_info arr_show_info[], size_t len_arr_show_info)

- **Назначение:** Запись данных из массива в файл, после удаления элементов.
- **Входные данные:** Аргументы коммандной строки, массив, хранящий в себе все записи, длина массива со всеми записями).
- **Выходные данные:** Код ошибки или код об успешном завершении работы функции.

int check_file(char **argv)

- **Назначение:** Проверка файла на корректность.
- **Входные данные:** Аргументы коммандной строки.
- **Выходные данные:** Код ошибки или код об успешном завершении работы функции.

void choose_menu(int *menu)

- **Назначение:** Обрабатка ввода номера пункта меню.
- **Входные данные:** Адрес на переменную, хрнящую значение выбора пункта меню.
- **Выходные данные:** Отсутствуют.

void print_elem_show(size_t id, char name_theater[], char name_show[], double start_price, double finish_price, char type_of_show[], char name_composer[], char country[], char type_of_musicial[], char start_age[], char duration[])

- **Назначение:** Вывод одной записи в формате таблицы.
- **Входные данные:** Индекс записи, название театра, название шоу, начальная цена, максимальная цена, жанр спектакля, имя композитора, название страны, музыкальный тип, минимальный возраст, продолжительность.
- **Выходные данные:** Отсутствуют.

void print_header_arr_show()

- **Назначение:** Вывод шапки таблицы для вывода всех записей.
- **Входные данные:** Отсутствуют.
- **Выходные данные:** Отсутствуют.

void print_elem_keys(size_t id, double start_price)

- **Назначение:** Вывод одной записи из массива структур с индексами и ключами.
- **Входные данные:** Индекс, ключ (начальная цена).
- **Выходные данные:** Отсутствуют.

void print_header_keys()

- **Назначение:** Вывод шапк таблицы для вывода таблицы индексов и ключей.
- **Входные данные:** Отсутствуют.
- **Выходные данные:** Отсутствуют.

void print_header_analysys()

- **Назначение:** Вывод шапки таблицы для вывода таблицы с результатами замеров.
- **Входные данные:** Отсутствуют.
- **Выходные данные:** Отсутствуют.

void print_elem_analisys(size_t len_arr, unsigned long long buble_sort_show, unsigned long long qsort_time_show, unsigned long long buble_sort_key, unsigned long long qsort_time_key)

- **Назначение:** Вывод результатов замера для одной длины массива.
- **Входные данные:** Длина массива, результат времени нашей сортировки массива структур со всеми данными, результат времени быстрой сортировки массива структур со всеми данными, результат времени нашей сортировки массива структур с индексами и ключами, результат времени быстрой сортировки массива структур с индексами и ключами.
- **Выходные данные:** Отсутствуют.

void print_elem_analisys_percent(size_t len_arr, double buble_sort_show, double qsort_time_show, double buble_sort_key, double qsort_time_key)

- **Назначение:** Вывод результатов процентного отношения времён замера сортировки массива одной длины.
- **Входные данные:** Длина массива, преимущество нашей сотрировки, относительно быстрой при сортировке массива со всеми данными, преимущество быстрой сотрировки, относительно нашей при сортировке массива со всеми данными, преимущество нашей сотрировки, относительно быстрой при сортировке массива с индексами и ключами, преимущество быстрой сотрировки, относительно нашей при сортировке массива с ключами и значениями.
- **Выходные данные:** Отсутствуют.

void change_elems_arr_struct_show(show_info arr_show_info[], size_t index_1, size_t index_2)

- **Назначение:** Перестановка двух элементов массива.
- **Входные данные:** Массив, хранящий все записи, индекс первого элемента, индекс второго элемента.
- **Выходные данные:** Отсутствуют.

void change_elems_arr_struct_keys(key_index arr_key_indexes[], size_t index_1, size_t index_2)

- **Назначение:** Перестановка двух элементов массива.

- **Входные данные:** Массив, хранящий структуры с индексом и ключом, индекс первого элемента, индекс второго элемента.
- **Выходные данные:** Отсутствуют.

int remove_element(show_info arr_show_info[], key_index arr_key_index[], size_t *len_arr_show_info)

- **Назначение:** Удаление элемента из массива.
- **Входные данные:** Массив, хранящий все записи, массив, хранящий структуры с индексом и ключом, длина массива.
- **Выходные данные:** Код ошибки или код об успешном завершении работы функции.

void setting_arr_key(key_index arr_key_index[], size_t len_arr_show_info)

- **Назначение:** Заполнение полей индексов в массиве с структурами, хранящими ключи.
- **Входные данные:** Массив, хранящий структуры с индексом и ключом, длина массива.
- **Выходные данные:** Отсутсвтуют.

int find_elem(show_info arr_show_info[], size_t len_arr_show_info)

- **Назначение:** Поиск элементов по заданным параметрам.
- **Входные данные:** Массив, хранящий все записи, длина массива.
- **Выходные данные:** Код ошибки или код об успешном завершении работы функции.

void print_all(show_info arr_show_info[], size_t len_arr_show_info)

- **Назначение:** Печать всей таблицы.
- **Входные данные:** Массив, хранящий все записи, длина массива.
- **Выходные данные:** Отсутствуют.

void sort_arr_show_info(show_info arr_show_info[], size_t len_arr_show_info)

- **Назначение:** Сортировка массива со всеми данными.
- **Входные данные:** Массив, хранящий все записи, длина массива.
- **Выходные данные:** Отсутствуют.

void sort_arr_key_index(key_index arr_key_index[], size_t len_arr_show_info)

- **Назначение:** Сортировка массива с структурами, хранящими индексы и ключи.
- **Входные данные:** Массив, хранящий структуры с индексами и ключами, длина массива.
- **Выходные данные:** Отсутствуют.

void print_one_elem(show_info arr_show_info[], size_t i)

- **Назначение:** Вывод эдного элемента таблицы.
- **Входные данные:** Массив, хранящий все записи, индекс элемента.
- **Выходные данные:** Отсутствуют.

void print_all_elems_with_keys(show_info arr_show_info[], key_index arr_key_index[], size_t len_arr_show_info)

- **Назначение:** Вывод всех таблиц при помощи таблицы ключей.
- **Входные данные:** Массив, хранящий все записи, массив, хранящий структуры с индексами и значениями, длина массива.
- **Выходные данные:** Отсутствуют.

int compare_index_keys(const void *a, const void *b)

- **Назначение:** Сравнение ключей в структурах.
- **Входные данные:** Указатель на элемент массива, указатель на элемент массива.
- **Выходные данные:** Код, показывающий результат сравнения.

int compare_show_arr(const void *a, const void *b)

- **Назначение:** Сравнение минимальных цен в структурах.
- **Входные данные:** Указатель на элемент массива, указатель на элемент массива.
- **Выходные данные:** Код, показывающий результат сравнения.

void analisys_time_sort_one_size(char **argv, key_index arr_key_index[], show_info arr_show_info[], size_t len_arr, size_t len_arr_show_info)

- **Назначение:** Вывод одной записи о результатах времени сортировки.

- **Входные данные:** Аргумент командной строки, массив структур с ключами и значениями, массив со всеми данными, длина сортируемого массива, длина изначального массива.
- **Выходные данные:** Отсутствуют.

int analysys_all_sizes(char **argv, key_index arr_key_index[], show_info arr_show_info[], size_t len_arr_show_info)

- **Назначение:** Вывод всех записей о результатах замеров сортирововк массивов разных длин.
- **Входные данные:** Аргумент командной строки, массив структур с ключами и значениями, массив со всеми данными, длина изначального массива.
- **Выходные данные:** Код ошибки или код об успешном завершении работы функции.

Набор тестов

Негативные:

№	Входные данные	Выходные данные	Класс эквивалентности
1	Ввод числа меньше 1	Ошибка выбора пункта меню	Проверка работы меню
2	Ввод числа больше 9	Ошибка выбора пункта меню	
3	Ввод большого числа	Ошибка выбора пункта меню	
4	Ввод маленького числа	Ошибка выбора пункта меню	
5	Ввод символа	Ошибка выбора пункта меню	
6	При добавлении элементов, вводится названия театра более размером в 25 символов	Ошибка при введениии названии театра!	Ошибка при введениии названии шоу при добавлении в файл
7	При добавлении элементов, вводится названия театра размером в 0 символов	Ошибка при введениии названии театра!	
8	При добавлении элементов, вводится названия спектакля размером боле 25 символов	Ошибка при введениии названии спектакля!	Ошибка при введениии названии спектакля при добавлении в файл
9	При добавлении элементов, вводится названия спектакля размером в 0 символов	Ошибка при введениии названии спектакля!	
10	При добавлении элементов, вводится минимальная цена билета в виде символа	Ошибка при введениии начальной цены!	Ошибка при введениии начальной цены при добавлении в файл
11	При добавлении элементов, вводится минимальная цена билета в отрицательного числа	Ошибка при введениии начальной цены!	
12	При добавлении элементов, вводится максимальная цена билета в виде символа	Ошибка при введениии конечной цены!	Ошибка при введениии конечной цены при добавлении в файл
13	При добавлении элементов, вводится максимальная	Ошибка при введениии конечной цены!	

	цена билета в отрицательного числа		
14	При добавлении элементов, вводится минимальная цена больше максимальной	Минимальная цена больше максимальной!	Минимальная цена больше максимальной при добавлении записи в файл
15	При удалении элементов, вводится названия спектакля размером более 25 символов	Ошибка введенного названия!	Ошибка введенного названия при удалении записей
16	При добавлении элементов, вводится названия спектакля размером в 0 символов	Ошибка введенного названия!	
17	При удалении элементов, вводится отрицательное количество элементов, которое мы хотим удалить	Ошибка ввода количества удалений при удалении записей	Ошибка ввода количества удалений при удалении записей
18	При удалении элементов, вводится символ, вместо количества записей, которое мы хотим удалить	Ошибка ввода количества удалений при удалении записей	
19	При вводе искомого значения, в пункте меню для выбора возраста введем число больше 3	Ошибка при введении возраста!	Ошибка ввода возраста при поиске записей
20	При вводе искомого значения, в пункте меню для выбора возраста введем число меньше 1	Ошибка при введении возраста!	
21	При вводе искомого значения, в пункте меню для выбора возраста введем число меньше 0	Ошибка при введении возраста!	
22	При вводе искомого значения, в пункте меню для выбора возраста введем символ	Ошибка при введении возраста!	
23	При поиске элементов, вводится отрицательная продолжительность спектакля	Ошибка при введении продолжительности!	Ошибка ввода продолжительности при поике записей

24	При поиске элементов, вводится символ, вместо продолжительности спектакля	Ошибка при введении продолжительности!	
25	При считывании элементов из файла, названия театра в длину более 25 символов	Ошибка при введениии названии театра!	Ошибка при введениии названия театра, при считывании из файла
26	При считывании элементов из файла, названия театра в длину равно 0 символов	Ошибка при введениии названии театра!	
27	При считывании элементов из файла, названия спектакля в длину более 25 символов	Ошибка при введениии названии спектакля!	Ошибка при введениии названия спектакля, при считывании из файла
28	При считывании элементов из файла, названия спектакля в длину равно 0 символов	Ошибка при введениии названии спектакля!	
29	При чтении элементов из файла в массив, минимальная цена билета представлена в виде символа	Ошибка при введениии начальной цены!	Ошибка при чтении начальной цены из файла в массив
30	При чтении элементов из файла в массив, минимальная цена билета представлена в виде отрицательного числа	Ошибка при введениии начальной цены!	
31	При чтении элементов из файла в массив, максимальная цена билета представлена в виде символа	Ошибка при введениии начальной цены!	Ошибка при чтении конечной цены из файла в массив
32	При чтении элементов из файла в массив, максимальная цена билета представлена в виде отрицательного числа	Ошибка при введениии конечной цены!	
33	При чтении элементов из файла в массив, в пункте меню для выбора жанра введем число больше 5	Ошибка при введении жанра!	Ошибка при считывании жанра из файла в массив
34	При чтении элементов из	Ошибка при введении	

	файла в массив, в пункте меню для выбора жанра введем число меньше 1	жанра!	
35	При чтении элементов из файла в массив, в пункте меню для выбора жанра введем число меньше 0	Ошибка при введении жанра!	
36	При чтении элементов из файла в массив, в пункте меню для выбора жанра введем символ	Ошибка при введении жанра!	
37	При считывании элементов из файла, имя композитора в длину более 25 символов	Ошибка при введениии имени композитора!	Ошибка при введениии имени композитора, при считывании из файла
38	При считывании элементов из файла, имя композитора в длину равно 0 символов	Ошибка при введениии имени композитора!	
39	При считывании элементов из файла, название страны в длину более 25 символов	Ошибка при введениии названия страны !	Ошибка при введениии названия страны, при считывании из файла
40	При считывании элементов из файла, название страны в длину равно 0 символов	Ошибка при введениии названия страны!	
41	При чтении элементов из файла в массив, в пункте меню для выбора музыкального жанра введем число больше 3	Ошибка при введении музыкального жанра!	Ошибка при считывании музыкального жанра из файла в массив
42	При чтении элементов из файла в массив, в пункте меню для выбора музыкального жанра введем число меньше 1	Ошибка при введении музыкального жанра!	
43	При чтении элементов из файла в массив, в пункте меню для выбора музыкального жанра введем число меньше 0	Ошибка при введении музыкального жанра!	
44	При чтении элементов из файла в массив, в пункте меню для выбора	Ошибка при введении музыкального жанра!	

	музыкального жанра введем символ		
45	При чтении элементов из файла в массив, в пункте меню для выбора возраста введем число больше 3	Ошибка при введении возраста!	Ошибка при считывании возраста из файла в массив
46	При чтении элементов из файла в массив, в пункте меню для выбора возраста введем число меньше 1	Ошибка при введении возраста!	
47	При чтении элементов из файла в массив, в пункте меню для выбора возраста введем число меньше 0	Ошибка при введении возраста!	
48	При чтении элементов из файла в массив, в пункте меню для выбора возраста введем символ	Ошибка при введении возраста!	
49	При чтении элементов из файла в массив, продолжительнось представлена в виде символа	Ошибка при введениии продолжительности!	Ошибка при чтении продолжительности из файла в массив
50	При чтении элементов из файла в массив, продолжительность представлена в виде отрицательного числа	Ошибка при введениии продолжительности!	
51	При вызове пункта вывода замеров времени сортировки, если в массие будет менее 1000 элементов	Нельзя запускать замеры, если элементов менее 1000!	Ошибка замеров времен сортировки
52	Добавление элемента в файл	Moscow Art Theater Anna Karenina 163.040000 4607.090000 1	Успешное добавление элемента в файл
53	Добавление элемента в файл	The Big Theater Nutcker 540.650000 5488.060000 5 Ivanox	

		Kazacstan 2 1 168	
54	Добавление элемента в файл	The Big Theater Gore ot Uma 321.590000 7561.430000 4 1	
55	Добавление элемента в файл	Moscow Art Theater Anna Karenina 483.060000 1904.140000 3	
56	Добавление элемента в файл	Vakhtangov Theater Revizor 410.830000 948.950000 2	
57	Чтение корректного файла	Файл в папке проекта	Успешное чтение файла
58	Сортировка корректного файла	Файл в папке проекта	Успешная сортировка записей
59	Удаление из корректного файла	Файл в папке проекта	Успешное удаление записей
60	Поиск в корректном файле	Файл в папке проекта	Успешная поиск записей
61	Замеры времени сортировок в корректном файле	Файл в папке проекта	Успешный замер времен сортировки записей
62	Сортировка массива структур с ключами в корректном файле	Файл в папке проекта	Успешная сортировка записей

Оценка времени сортировки

Алгоритм замера времени работает так:

- Берётся таблица записей и соответствующая ей таблица с индексами и ключами;
- Таблицы сортируются по отдельности при помощи сортировки пузырьком. При этом производится замер времени сортировок;
- Замеры повторяются 20 раз. Полученное суммарное время сортировок делится на 20;
- Замеры проводятся так же и для быстрой сортировки.

Результаты замеров:

Количесвто элементов	Сортировка пузырьком массива структур со всеми данными в тиках	Быстрая сортировка массива структур со всеми данными в тиках	Сортировка пузырьком массива структур с ключами в тиках	Быстрая сортировка массива структур с ключами в тиках
5	3351	5080	2670	4743
Процентное соотношение				
5	+51.59%	-34.03%	+77.64%	-43.71%

При использовании быстрой сортировки для сортировки массива структур с ключами можно выиграть +6.63% времени, затратив на 11.11% памяти больше.

При использовании пузырьковой сортировки для сортировки массива структур с ключами можно выиграть +20.32% времени, затратив на 11.11% памяти больше.

Количесвто элементов	Сортировка пузырьком массива структур со всеми данными в тиках	Быстрая сортировка массива структур со всеми данными в тиках	Сортировка пузырьком массива структур с ключами в тиках	Быстрая сортировка массива структур с ключами в тиках
50	275233	69155	198241	64822
Процентное соотношение				
50	-74.87%	+297.99%	-67.30%	+205.82%

При использовании быстрой сортировки для сортировки массива структур с ключами можно выиграть +6.26% времени, затратив на 11.11% памяти больше.

При использовании пузырьковой сортировки для сортировки массива структур с ключами можно выиграть +27.97% времени, затратив на 11.11% памяти больше.

Количесвто элементов	Сортировка пузырьком массива структур со всеми данными в тиках	Быстрая сортировка массива структур со всеми данными в тиках	Сортировка пузырьком массива структур с ключами в тиках	Быстрая сортировка массива структур с ключами в тиках
100	1034701	173163	792960	160377
Процентное соотношение				
100	-83.26%	+497.53%	-79.77%	+394.43%

При использовании быстрой сортировки для сортировки массива структур с ключами можно выиграть +7.38% времени, затратив на 11.11% памяти больше.

При использовании пузырьковой сортировки для сортировки массива структур с ключами можно выиграть +23.36% времени, затратив на 11.11% памяти больше.

Количесвто элементов	Сортировка пузырьком массива структур со всеми данными в тиках	Быстрая сортировка массива структур со всеми данными в тиках	Сортировка пузырьком массива структур с ключами в тиках	Быстрая сортировка массива структур с ключами в тиках
500	25811615	1144761	19482584	1029532
Процентное соотношение				
500	-95.56%	+2154.76%	-94.72%	+1792.37%

При использовании быстрой сортировки для сортировки массива структур с ключами можно выиграть +10.07% времени, затратив на 11.11% памяти больше.

При использовании пузырьковой сортировки для сортировки массива структур с ключами можно выиграть +24.52% времени, затратив на 11.11% памяти больше.

Количесвто элементов	Сортировка пузырьком массива структур со всеми данными в тиках	Быстрая сортировка массива структур со всеми данными в тиках	Сортировка пузырьком массива структур с ключами в тиках	Быстрая сортировка массива структур с ключами в тиках
1000	101553054	2550468	75914250	2267890
Процентное соотношение				
1000	-97.49%	+3881.74%	-97.01%	+3247.35%

При использовании быстрой сортировки для сортировки массива структур с ключами можно выиграть +11.08% времени, затратив на 11.11% памяти больше.

При использовании пузырьковой сортировки для сортировки массива структур с ключами можно выиграть +25.25% времени, затратив на 11.11% памяти больше.

Вывод

- Тип данных "запись" с вариантной частью целесообразно использовать, когда обрабатываемые данные имеют как общие, так и различные поля. Наибольшая выгода достигается в тех случаях, когда размеры вариантных частей для разных типов данных примерно одинаковы. В противном случае, при значительном различии размеров, объем неиспользованной памяти может быть существенным.
- Основное преимущество использования дополнительного массива ключей заключается в сокращении времени сортировки. Например, сортировка пузырьком ускоряется в среднем в 1.3 раза, а быстрая сортировка ускоряется в среднем в 1.2 раза. Это происходит потому, что объем данных, которые необходимо сортировать в массиве ключей, значительно меньше, чем в полном массиве записей: в таблице ключей нужно обменивать лишь два поля, тогда как в таблице записей — все. Кроме того, более эффективные алгоритмы сортировки, такие как quick_sort, дают большее ускорение по сравнению с менее эффективными, например, сортировкой пузырьком, что позволяет ускорить процесс сортировки еще больше.

Ответы на вопросы

- **Как выделяется память под вариантную часть записи?**
  Память для вариантной части выделяется в объеме, соответствующем максимальному размеру одного из полей этой части.
- **Что будет, если в вариантную часть ввести данные, несоответствующие описанным?**
  Если в вариантную часть будут введены данные, не соответствующие описанным, и при этом отсутствуют проверки со стороны программиста, это приведет к неопределенному поведению программы.
- **Кто должен следить за правильностью выполнения операций с вариантной частью записи?**
  За корректное выполнение операций с вариантной частью записи отвечает программист, который должен предусмотреть необходимые проверки и ограничения.
- **Что представляет собой таблица ключей, зачем она нужна?**
  Таблица ключей — это массив, состоящий из структур, каждая из которых включает индекс записи в исходной таблице и значение интересующего поля. Такая таблица позволяет ускорить сортировку и выполнение других операций, уменьшая объем данных для обработки.
- **В каких случаях эффективнее обрабатывать данные в самой таблице, а когда – использовать таблицу ключей?**
  Обрабатывать данные в самой таблице эффективнее, если объем данных незначителен, так как это экономит память. Когда разница между размерами данных в таблице ключей и таблице записей становится значительной, обработка через таблицу ключей становится более эффективной.
- **Какие способы сортировки предпочтительнее для обработки таблиц и почему?**
  Для сортировки таблиц лучше использовать высокоэффективные алгоритмы с временной сложностью O(nlog(n)) или меньшей, а также такие алгоритмы, которые минимизируют количество перестановок.