

**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**

### Студент Завойских Евгения Васильевна

*фамилия, имя, отчество*

### Группа ИУ7-33Б

Студент Завойских Е.В.

*подпись, дата фамилия и.о.*

### Руководитель практики Барышникова М.Ю.

*подпись, дата фамилия, и.о.*

### Оценка

*2021 г.*

**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 репертуар театров, содержащий: название театра, спектакль, режиссер, диапазон цены билета, тип спектакля: детский – для какого возраста, тип (сказка, пьеса); взрослый – (пьеса, драма, комедия); музыкальный – (композитор, страна, минимальный возраст, продолжительность). Вывести список всех музыкальных спектаклей для детей указанного возраста с продолжительностью меньше указанной.

**Внешняя спецификация программы**

1. Входные данные

При вызове программы вводится название файла, откуда считываются записи о спектаклях. Каждое поле записи в файле расположено на отдельной строке, тип спектакля представлен в виде числа (детский – 1, взрослый – 2, музыкальный – 3), тип детского спектакля – так же в виде числа (сказка – 1, пьеса – 2), тип взрослого – в виде числа (пьеса – 1, драма – 2, комедия – 3).

При добавлении записи в таблицу считывание происходит с консоли, ввод полей - как в файле.

Для выбора дальнейшего действия каждый раз выводится меню, пользователь вводит число, соответствующее нужному пункту меню.

При удалении записи вводится нужный индекс.

1. Выходные данные

При сортировке таблицы ключей или спектаклей выводится соответствующая таблица с подписанными столбцами; результаты использования алгоритмов сортировок так же выводятся с пояснениями; при добавлении и удалении записи изменяется и сам файл. В случае некорректного ввода команды меню или записи выводится сообщение об ошибке и предоставляется возможность выбрать команду еще раз. При ошибке чтения файла выводится сообщение об ошибке, и программа завершается.

1. Задача, реализуемая программой

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

1. Добавление записи в таблицу.
2. Удаление записи из таблицы.
3. Вывод всех музыкальных спектаклей для детей указанного возраста с продолжительностью меньше указанной.
4. Вывод отсортированной таблицы ключей при неотсортированной исходной.
5. Вывод отсортированной таблицы спектаклей.
6. Вывод отсортированной таблицы спектаклей с применением отсортированной таблицы ключей.
7. Вывод результатов различных сортировок двух таблиц (по времени).
8. Вывод исходной таблицы спектаклей.

1. Способ обращения к программе

Программа может быть вызвана из консоли или запущена в любой среде разработки, поддерживающей язык С.

1. Возможные аварийные ситуации и ошибки пользователя

В случае аварийной ситуации программа выводит сообщение о произошедшей ошибке. Ошибка может возникнуть в результате некорректного ввода (не соответствующего формату), а так же при отсутствии записей в файле, присутствии некорректных записей в файле и переполнении массива записей.

**Внутренние структуры данных**

В программе используется структура с вариантной частью для хранения записи о спектакле. Структура удобна в данном случае, так как нам нужно хранить несколько полей о спектакле, вариантная часть нужна, чтобы была возможность выбирать один тип спектакля из нескольких.

Для описания детского типа и музыкального так же используются структуры.

// максимальные размеры строк

#define THEATRE\_NAME\_LEN 15

#define PERF\_NAME\_LEN 15

#define DIR\_NAME\_LEN 15

#define COMPOSER\_NAME\_LEN 15

#define COUNTRY\_NAME\_LEN 15

// перечисление типов спектакля

typedef enum {CHILD = 1, ADULT, MUSIC} perf\_type\_num\_e;

// перечисление для определения типа детского спектакля

typedef enum {FAIRY\_TALE = 1, CHILD\_PLAY} child\_perf\_type\_e;

// перечисление для определения типа взрослого спектакля

typedef enum {ADULT\_PLAY = 1, DRAMA, COMEDY} adult\_perf\_e;

// структура для определения детского типа спектакля

struct child\_perf

{

    short age; // возраст

    child\_perf\_type\_e type; // тип

};

typedef struct child\_perf child\_perf\_s;

// структура для определения музыкального типа спектакля

struct musical\_perf

{

    char composer[COMPOSER\_NAME\_LEN + 1]; // композитор

    char country[COUNTRY\_NAME\_LEN + 1]; // страна

    short min\_age; // минимальный возраст

    short duration; // продолжительность

};

typedef struct musical\_perf musical\_perf\_s;

// объединение для определения содержимого типа спектакля

union type\_content

{

    child\_perf\_s child; // детский

    adult\_perf\_e adult; // взрослый

    musical\_perf\_s music; // музыкальный

};

typedef union type\_content type\_content\_u;

// структура для определения типа спектакля

struct perf\_type

{

    perf\_type\_num\_e num; // номер типа

    type\_content\_u cont; // содержимое типа

};

typedef struct perf\_type perf\_type\_s;

// структура для определения одного спектакля в конкретном театре

struct performance

{

    char theatre\_name[THEATRE\_NAME\_LEN + 1]; // театр

    char performance\_name[PERF\_NAME\_LEN + 1]; // название спектакля

    char director[DIR\_NAME\_LEN + 1]; // режиссер

    short price\_range[2]; // диапазон цены билета

    perf\_type\_s type; // тип спектакля

};

typedef struct performance performance\_s;

Для описания строки таблицы ключей используется структура, содержащая индекс записи из исходной таблицы и ключевое поле (минимальная цена билета).

// структура для записи в таблице ключей (ключ - минимальная цена билета)

struct perf\_min\_price

{

    int index; // индекс записи в исходном массиве

    short min\_price; // минимальная цена билета

};

typedef struct perf\_min\_price perf\_min\_price\_s;

**Алгоритм**

Считываются все записи из файла (каждое поле – на отдельной строке). Далее выполняется действие, выбранное из меню:

1. Добавление записи: запись считывается с консоли, затем она добавляется в массив записей, далее – в файл
2. Удаление записи: считывается индекс, соответствующая запись удаляется из массива записей (если есть такой индекс), затем – из файла
3. Вывод всех музыкальных спектаклей для детей указанного возраста с продолжительностью меньше указанной: считывается возраст и длительность, затем в массиве ищутся нужные записи и выводятся в виде таблицы на экран
4. Вывод отсортированной таблицы ключей при не отсортированной исходной: из файла считывается массив записей, на его основе формируется массив ключей, этот массив сортируется по значению ключа и выводится в виде таблицы
5. Вывод отсортированной таблицы спектаклей: массив сортируется и выводится на экран в виде таблицы
6. Вывод отсортированной таблицы спектаклей с применением отсортированной таблицы ключей: из файла считывается массив записей, на его основе формируется массив ключей, этот массив сортируется по значению ключа, выводится таблица спектаклей по индексам отсортированной таблицы ключей
7. Вывод результатов различных сортировок двух таблиц (по времени): в цикле из файла считывается массив записей и замеряется время его сортировки сначала пузырьком, потом быстрой сортировкой, аналогично происходит с массивом ключей, выводятся результаты в консоль
8. Вывод исходной таблицы спектаклей: из файла считывается массив записей, они выводятся в виде таблицы

**Тесты**

Негативы:

|  |  |  |
| --- | --- | --- |
| Входные данные | Вывод | Что проверяется |
| Пустой файл | Не удалось считать записи из файла | Пустой файл |
| Файл не существует | Не удалось открыть указанный файл | Неверное имя файла |
| new theatre  new perf aaaaaaaaa  new director  300  1500  3  new composer  new country  12  100 | Не удалось считать записи из файла | Название превышает 15 символов |
| Введите название театра: a  Введите название спектакля: a  Введите имя режиссера: a  Введите минимальную цену билета: a | Ошибка ввода | Неверно введено числовое поле |
| Введите название театра: a  Введите название спектакля: a  Введите имя режиссера: a  Введите минимальную цену билета: 3  Введите максимальную цену билета: 4  Для детского типа спектакля вводится возраст и тип: 1 - сказка, 2 - пьеса  Для взрослого типа спектакля вводится тип: 1 - пьеса, 2 - драма, 3 - комедия  Для музыкального типа спектакля вводится композитор, страна, минимальный возраст, продолжительность  Каждый пункт вводится в отдельной строке  Введите тип спектакля: 1 - детский, 2 - взрослый, 3 - музыкальный: 0 | Ошибка ввода | Введен несуществующий тип |
| Введите название театра: a  Введите название спектакля: a  Введите имя режиссера: a  Введите минимальную цену билета: -10 | Ошибка ввода | Числовое поле отрицательно |
| 9 | Нет команды с введенным номером | Нет команды с введенным номером |
| Введите индекс удаляемой записи: 10 | Ошибка ввода | Нет записи с указанным индексом |
| Файл с 50-ю записями, попытка добавить запись | Таблица уже заполнена | Добавление в уже заполненный массив |

Позитивы:

|  |  |  |
| --- | --- | --- |
| Входные данные | Вывод | Что проверяется |
| 1  Введите название театра: a  Введите название спектакля: a  Введите имя режиссера: a  Введите минимальную цену билета: 10  Введите максимальную цену билета: 100  Для детского типа спектакля вводится возраст и тип: 1 - сказка, 2 - пьеса  Для взрослого типа спектакля вводится тип: 1 - пьеса, 2 - драма, 3 - комедия  Для музыкального типа спектакля вводится композитор, страна, минимальный возраст, продолжительность  Каждый пункт вводится в отдельной строке  Введите тип спектакля: 1 - детский, 2 - взрослый, 3 - музыкальный: 1  10  2 | 1.PNG | Добавление записи в файл |
| 2  Введите индекс удаляемой записи: 40 | 2.PNG | Удаление записи из файла |
| 3  Введите возраст и продолжительность: 17 200 | 3.PNG | Поиск музыкальных спектаклей |
| 4 | |perf. table ind |min. price  --------------------------  |22 |100  |5 |200  |11 |240  |19 |300  |1 |300  |2 |500  |17 |1000  |0 |1000  |14 |1000  |32 |1000  |20 |1000  |7 |1000  |23 |1000  |6 |1100  |38 |1200  |30 |1200  |25 |1230  |36 |1299  |3 |1300  |31 |1300  |16 |1500  |24 |1550  |28 |2000  |12 |2000  |21 |2000  |26 |2300  |9 |2300  |29 |2900  |39 |3000  |10 |3000  |18 |3000  |34 |3002  |27 |3300  |35 |3400  |33 |3400  |37 |3999  |13 |4000  |8 |4000  |15 |5000  |4 |5000 | Вывод отсортированной таблицы ключей |
| 5 | 4.PNG | Вывод отсортированного массива записей |
| 6 | 4.PNG | Вывод отсортированных записей по отсортированному массиву ключей |
| 7 | Время пузырьковой сортировки исходной таблицы: 0.037333  Время быстрой сортировки исходной таблицы: 0.014667  Время пузырьковой сортировки таблицы ключей: 0.010667  Время быстрой сортировки таблицы ключей: 0.005333 | Замеры времени работы различных сортировок с каждой из таблиц |

**Вывод**

Замеры времени (в тактах) в массивах, содержащих по 40 элементов, при 1500-кратном повторении (приведены усреднённые результаты):

|  |  |  |
| --- | --- | --- |
|  | Время сортировки таблицы ключей | Время сортировки таблицы спектаклей |
| Пузырьковая | 0.010667 | 0.037333 |
| Быстрая | 0.005333 | 0.014667 |

Сортировка таблицы ключей работает быстрее сортировки всей таблицы на 28% при сортировке пузырьком и на 36% - при быстрой сортировке.

Память, необходимая для хранения таблицы из 40 записей: 88 \* 40 = 3520 байт. Память, необходимая для хранения таблицы ключей: 6 \* 40 = 240 байт, что составляет всего около 7% от памяти, выделяемой под таблицу записей.

Сортировка таблицы ключей работает намного быстрее, чем сортировка таблицы записей; память, выделяемая для хранения таблицы ключей, составляет менее десятой доли от памяти под таблицу записей, так что в этой задаче нам выгодно использовать массив ключей для работы с массивом записей.

**Ответы на вопросы**

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

Вариантная часть может быть реализована с помощью объединения. Объединение может иметь несколько полей, количество памяти, выделяемое под объединение, равно размеру максимального по размеру поля, так как все поля занимают одну область памяти (начинаются с одной ячейки памяти).

1. **Что будет, если в вариантную часть ввести данные, несоответствующие описанным?** Тип данных не проверяется при компиляции, поэтому данные будут записаны, но это повлечёт ошибки в работе программы.
2. **Кто должен следить за правильностью выполнения операций с вариантной частью записи?**

Программист.

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

Это таблица, содержащая индекс элемента в исходной таблице и значение выбранного ключевого поля элемента. Таблица ключей экономит время обработки основной таблицы (например, при сортировке по ключевому полю).

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

Если исходная таблица небольшого размера, то эффективнее обрабатывать её сразу всю. Однако, если же массив данных большой, то эффективнее использовать таблицу ключей.

1. **Какие способы сортировки предпочтительнее для обработки таблиц и почему?**

Предпочтительнее те способы сортировки, которые не трогают уже отсортированные элементы, так как перестановка элементов таблицы отнимает еще время.