|  |  |
| --- | --- |
|  | **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 К ЛАБОРАТОРНОЙ РАБОТЕ №2 ПО «ТИПАМ И СТРУКТУРАМ ДАННЫХ»**

**Тема: «Записи с вариантами. Обработка таблиц»**

Группа: ИУ7-32Б

|  |  |  |
| --- | --- | --- |
| Студент: |  | Сидоров Максим Михайлович |

Преподаватель: Никульшина Татьяна Александровна

**Цель работы:** научиться работать с типом данных **struct** и **union**, а также записями, содержащими вариантную часть. Сравнить эффективность сортировок с использованием исходной таблицы и таблицы ключей, а также разных алгоритмов сортировки (сортировка пузырьком и qsort).

**Условие задачи:** создать таблицу, содержащую не менее 40 записей театров с вариантной частью. Произвести поиск информации по мюзиклам, удовлетворяющим указанному возрасту и длительности. Упорядочить таблицу, по возрастанию ключей (где ключ – минимальная цена билета), используя: а) исходную таблицу; б) массив ключей, используя 2 разных алгоритма сортировки. Оценить эффективность этих алгоритмов (по времени и по используемому объему памяти) в случаях а) и б). Обосновать выбор алгоритма сортировки. Оценка эффективности должна быть относительной (в %).

Запись театра содержит: название театра, спектакль, режиссера, диапазон цены билета, тип спектакля: детский – для какого возраста, тип (сказка, пьеса); взрослый – пьеса, драма, комедия); музыкальный – композитор, страна, минимальный возраст, продолжительность).

**Входные данные:** файл с данными, номер команды из меню, ручной ввод.

Команды:

1. Вывести исходную таблицу
2. Добавить запись в конец таблицы
3. Удалить запись по названию спектакля
4. Вывести таблицу ключей
5. Отсортировать по минимальной цене билета
6. Вывести список музыкальных спектаклей
7. Вывести сортируемую таблицу
8. Сравнение эффективности при работе с таблицей и таблицей ключей
9. Сравнение эффективности сортировок
10. Вывод по таблице ключей
11. Выйти

**Выходные данные:** в зависимости от выбранной команды программа может вывести:

- таблицу

- таблицу ключей

- таблицу по ключам

- результат выборки

- сравнение эффективности

**Аварийные ситуации:**

- ввод неправильной команды

- пустой файл

- удаление несуществующего элемента

- неправильный ввод полей

**Описание СД:**

1. Для хранения информации о типах спектаклей и для кого они предназначены было выбрано следующее представление:

typedef **enum** {CHILD, ADULT, MUSICAL} show\_kind;

typedef **enum** {FAIRY\_TAIL, PLAY, DRAMA, COMEDY} type;

*Листинг 1. Представление перечисляемых типов.*

1. Для хранения информации о детских спектаклях было выбрано следующее представление:

typedef struct

{

int min\_age;

type type;

} child;

*Листинг 2. Представление информации о детском спектакле.*

1. Для хранения информации о взрослых спектаклях было выбрано следующее представление:

typedef struct

{

type type;

} adult;

*Листинг 3. Представление информации о взрослом спектакле.*

1. Для хранения информации о мюзиклах было выбрано следующее представление:

typedef struct

{

char composer[21];

char country[21];

int min\_age;

int duration;

} musical;

*Листинг 4. Представление информации о мюзикле.*

1. Для хранения информации о театрах было выбрано следующее представление:

typedef struct

{

char name[21];

char show[21];

char director[21];

int min\_ticket\_price;

int max\_ticket\_price;

show\_kind show\_kind;

union

{

child child; *// 1*

adult adult; *// 2*

musical musical; *// 3*

} kind;

} theatre;

*Листинг 4. Представление информации о мюзикле.*

**Функции:**

void print\_table(theatre a[], int n) - вывод таблицы на экран

void print\_data\_table(void) – вывод исходной таблицы на экран

void print\_menu(void) - вывод меню

void print\_main(theatre \*a) - вывод невариантных полей на экран

void print\_adult(theatre \*a) - вывод вариантных полей для взрослых спектаклей на экран

void print\_musical(theatre \*a) - вывод вариантных полей для мюзиклов на экран

void print\_child(theatre \*a) - вывод вариантных полей для детских спектаклей на экран

void print\_theatre(theatre \*a) - вывод информации о театре на экран

void print\_musical\_by\_age\_duration(theatre arr[], int n, int age, int duration) - вывод мюзиклов, соответствующих условию на экран

void print\_key\_table(key a[], int n) - вывод таблицы ключей на экран

int read\_theatre\_stdin(theatre \*a) - считывание записи из терминала

int read\_theatre(theatre \*a, FILE \*f) - считывание записи из файла

int put\_main(theatre \*a, const char \*name, const char \*show, const char \*director, int min\_price, int max\_price, int type) - сохранение невариантной части записи

int put\_child(theatre \*a, int age, int type) - сохранение вариантной части записи детского спектакля

int put\_adult(theatre \*a, int age, int type) - сохранение невариантной части записи взрослого спектакля

int put\_musical(theatre \*a, int age, int type) - сохранение невариантной части записи мюзикла

void file\_write(theatre \*a, int n) - запись в файл

void create\_key\_array(key a[], int n, theatre src[]) - создание таблицы ключей

void delete\_by\_pos(key a[], theatre b[], int \*n, int pos) - удаление записи по позиции

void sort(unsigned long \*time\_key, unsigned long \*time\_theatre\_qsort, unsigned long \*time\_theatre\_bubble, theatre arr[], key a[], int n) - сортировка различными способами и алгоритмами, а также замер времени

void my\_sort(theatre arr[], int n) - сортировка пузырьком

void swap(theatre \*a, theatre \*b) - обмен значений

int comp\_theatre(const void \*a, const void \*b)

- компаратор для сравнения таблицы

int comp\_key(const void \*a, const void \*b)

- компаратор для сравнения таблицы ключей

**Тестовые данные:**

|  |  |  |
| --- | --- | --- |
| **Описание** | **Вход** | **Выход** |
| Некорректный файл | Hgsd.txt | Не удалось открыть файл |
| Пустой файл | a.txt | Пустой файл |
| Неверная команда | F, d, -2 | Некорректный ввод |
| Неверный ввод записи | Comp, show, teatr, a, s | Некорректный ввод |
| Слова >20 символов | qwertyuiopasdfghjklasdfghj | Некорректный ввод |
| Сортировка | - | Отсортированная таблица |
| Вывод по таблице ключей | - | Отсортированная таблица |
| Добавление записи | A, aa, aaa, 2, 3, 2, 2 | Запись в таблицу |
| Удаление по названию шоу | Chess | Удаление записи |
| Удаление по названию шоу | hgsdfhjgsdfg | Нет такого шоу |
| Вывод исходной таблицы | - | Вывод исходной таблицы |
| Вывод сортируемой таблицы | - | Вывод отсортированной таблицы |
| Вывод времени сортировок | - | Время для bubble\_sort и qsort |

**Оценка эффективности:**

В качестве обычной сортировки была выбрана сортировка пузырьком, ее сложность – О(n^2).

Сложность быстрой сортировки – О(n\*log(n))

*Таблица 1. Время выполнения алгоритмов в миллисекундах.*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Количество записей** | **Bubble sort** | **Qsort** | **Bubble sort keys** | **Qsort keys** |
| **40** | 4 | 2 | 1 | 0 |
| **120** | 182 | 16 | 42 | 7 |
| **800** | 1017 | 66 | 235 | 29 |

Размер записи таблицы – 128 байт, а ключа – 8 байт.

|  |  |  |
| --- | --- | --- |
| **Количество записей** | **Таблица** | **Ключи** |
| **40** | 5120 | 5440 |
| **120** | 15360 | 16320 |
| **800** | 102400 | 108800 |

*Таблица 2. Количество занимаемой памяти в байтах.*

Из таблицы можно увидеть, что быстрая сортировка быстрее сортировки пузырьком на 91,1%.

Сортировка таблицы ключей быстрее сортировки всей таблицы на 56,6%.

Но затраты памяти при использовании таблицы ключей увеличилось на 6,25%.

**Вывод:** для работы с данными, содержащими вариативную часть, наиболее удобна структура с объединением. В ходе тестирования было выявлено, что быстрая сортировка эффективнее сортировки пузырьком, что использование таблицы ключей значительно быстрее, чем использование исходной таблицы, хотя она незначительно проигрывает по используемой памяти.

**Ответы на вопросы:**

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

Размер памяти равен памяти, занимаемой максимальным по длине ssполем. Эта память является общей для всех полей вариантной части.

1. **Что будет, если в вариантную часть ввести данные, не соответствующие описанным?**

Неопределенное поведение.

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

Разработчик.

1. **Что представляет собой таблица ключей? Зачем она нужна?**

Таблица ключей представляет из себя таблицу, где записаны индекс в исходной таблице и сортируемое значение (ключ). После сортировки таблицы ключей, мы можем обратится к исходной таблице по индексу элемента.

При большом количестве записей в исходной таблице использование таблицы ключей может уменьшить время обработки данных.

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

Обрабатывать исходную таблицу следует, если количество свободной памяти ограничено или когда размер таблицы небольшой.

Использовать таблицу ключей следует, если нужно сэкономить время обработки данных и поля для сортировки занимают много меньшую память по сравнению с основной таблицей.

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

Быстрее работает та сортировка, которая делает меньше перестановок.