|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **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» (ИУ) |
| Кафедра | «Программное обеспечение ЭВМ и информационные технологии» (ИУ7) |

**Лабораторная работа №2**

**“Записи с вариантами. Обработка таблиц”**

**Вариант №8**

|  |  |
| --- | --- |
| Студент: Князев Дмитрий Юрьевич, группа ИУ7-33Б | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) |
| Преподаватель:  Силантьева Александра Васильевна | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) |

Оглавление

Условия задачи 3

Входные и выходные данные 4

Описание структур данных 5

Оценка эффективности 7

Особенности реализации 9

Вывод 9

Контрольные вопросы 10

**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 – название театра), используя:

а) саму таблицу,

б)массив ключей.

Добавить возможность удаления записей по номеру и добавления новых записей в конец таблицы в ручном режиме

Добавить возможность вывода списка всех музыкальных спектаклей для детей указанного возраста с продолжительностью меньше указанной.

Репертуар театров содержит:

* название театра,
* спектакль,
* режиссёр,
* диапазон цены билета,
* тип спектакля:
  + - детский
      * + для какого возраста
        + тип (сказка, пьеса);
    - взрослый
      * + тип (пьеса, драма, комедия)
    - музыкальный
      * + композитор
        + страна
        + минимальный возраст
        + продолжительность

**Входные данные:**

* Пункт меню (число от 0 до 9 включительно)
* В зависимости от выбранного пункта меню:
  + Файл с данными о театрах, поля расположены построчно, разделителем записей является пустая строка
  + Данные о театре, вводимые через консоль

**Выходные данные (в зависимости от выбранного пункта меню):**

* Таблица записей
* Отсортированная по названию театра таблица записей (с использованием массива ключей или без)
* Отсортированный по названию театра массив ключей
* Данные о музыкальных спектаклях для детей указанного возраста с продолжительностью меньше указанной
* Сравнение эффективности двух алгоритмов сортировки с использованием массива ключей и без

**Функции меню**

0. Выход

1. Загрузить список театров из файла в таблицу

2. Добавить запись в конец таблицы

3. Вывести таблицу

4. Удалить запись по номеру

5. Вывести список всех музыкальных спектаклей для детей указанного возраста с продолжительностью меньше указанной

6. Вывести массив ключей, отсортированный по названию театра

7. Вывести таблицу, отсортированную по названию театра

8. Вывести таблицу, отсортированную по названию театра, используя массив ключей

9. Сравнить эффективность двух алгоритмов сортировки с использованием массива ключей и без

**Описание структур данных:**

|  |
| --- |
| **#define SHORT\_STRING\_MAX\_LENGTH 40**  **typedef enum {ERR\_TYPE, FAIRY\_TALE, PLAY, DRAMA, COMEDY} play\_type\_t;**  **typedef** **enum** {ERR\_KIND, CHILD, ADULT, MUSICAL} **show\_kind\_t**;  **typedef** **struct** record  {  **char** theater\_name[SHORT\_STRING\_MAX\_LENGTH + **2**];  **char** \*theater\_name\_long;  **char** play\_name[SHORT\_STRING\_MAX\_LENGTH + **2**];  **char** \*play\_name\_long;  **char** producer[SHORT\_STRING\_MAX\_LENGTH + **2**];  **char** \*producer\_long;  **uint64\_t** price\_range;  **show\_kind\_t** show\_kind;  **play\_type\_t** play\_type;  **union**  {  **struct** child  {  **char** age;  } child;  **struct** musical  {  **char** composer[SHORT\_STRING\_MAX\_LENGTH + **2**];  **char** \*composer\_long;  **char** country[SHORT\_STRING\_MAX\_LENGTH + **2**];  **char** \*country\_long;  **char** min\_age;  **int** duration;  } musical;  } play\_group;  } **record\_t**; |

Структура записи, хранит поля соответствующие описанным в условии задачи, деля текстовые поля на два типа: короткие (размером не более SHORT\_STRING\_MAX\_LENGTH байт), хранящиеся рядом с другими полями записи и длинные (размером больше, чем SHORT\_STRING\_MAX\_LENGTH байт), которые хранятся в виде указателей на объекты динамической памяти.

Информация о типе спектакля и типе представления хранится в переменных перечислимого типа show\_kind и play\_type.

Поле price\_range содержит информацию о диапазоне цен, используя дополнительные функции для установки и считывания значений.

|  |
| --- |
| **typedef** **struct** key  {  **char** theater\_name[SHORT\_STRING\_MAX\_LENGTH + **2**];  **char** \*theater\_name\_long;  **size\_t** id;  } **key\_t**; |

Структура ключа, хранит идентификатор записи из таблицы, копию короткого имени театра и указатель на длинное имя театра исходной записи.

|  |
| --- |
| **typedef** **struct** record\_table  {  **record\_t** \*records;  **key\_t** \*keys;  **size\_t** max\_size;  **size\_t** size;  } **record\_table\_t**; |

Структура таблицы, хранит указатель на массив записей, указатель на массив ключей, максимальный и текущий размеры.

**Оценка эффективности**

Оценка эффективности скорости сортировки пузырьком (средние значения ста измерений)

|  |  |  |  |
| --- | --- | --- | --- |
| Количество записей | Время без использования ключей в микросекундах | Время с использованием ключей в микросекундах | Отношение времени работы с ключами к времени работы без ключей |
| 10 | 13 | 3 | 433% |
| 50 | 327 | 73 | 447% |
| 100 | 1171 | 292 | 401% |
| 500 | 36126 | 7220 | 500% |
| 1000 | 197417 | 39734 | 496% |
| 5000 | 4843972 | 1010241 | 479% |

Оценка эффективности скорости быстрой сортировки (средние значения ста измерений)

|  |  |  |  |
| --- | --- | --- | --- |
| Количество записей | Время без использования ключей в микросекундах | Время с использованием ключей в микросекундах | Отношение времени работы с ключами к времени работы без ключей |
| 10 | 1 | 1 | 100% |
| 50 | 2 | 2 | 100% |
| 100 | 5 | 5 | 100% |
| 500 | 50 | 47 | 106% |
| 1000 | 115 | 107 | 107% |
| 5000 | 750 | 639 | 117% |

Оценка эффективности использования памяти

|  |  |  |  |
| --- | --- | --- | --- |
| Количество элементов | Размер массива ключей в байтах | Размер таблицы записей в байтах | Отношение размера массива ключей к размеру таблицы |
| 10 | 640 | 3455 | 18% |
| 50 | 3200 | 16457 | 19% |
| 100 | 6400 | 32914 | 19% |
| 500 | 32000 | 164570 | 19% |
| 1000 | 64000 | 1645700 | 19% |
| 5000 | 320000 | 1645700 | 19% |

**Особенности реализации**:

* Вводить тип спектакля и представления можно в любом регистре.
* В качестве ввода поддерживается любые символы Unicode.
* Некорректный ввод не завершает работу программы, а лишь возвращает пользователя в главное меню.
* После вывода таблиц стирается история выполнения команд программы

**Вывод:**

Объединения помогают экономить память, когда в массиве требуется хранить записи с вариантной частью, но следует помнить, что ответственность за правильное использование вариантной части несёт программист.

Использование массива ключей, состоящих из небольшого количества полей, ускоряет сортировку таблицы с большим количеством полей за счёт использования дополнительной памяти. Особенно заметен выигрыш в скорости при использовании алгоритма с квадратичной сложностью по времени.

Оценка эффективности на основе 100 итераций сортировки 5000 записей:

Не используя массив ключей, сортировка пузырьком заняла 4843972 микросекунд, быстрая сортировка — 750 микросекунд, то есть при выборе оптимального алгоритма, время работы сокращается в 6458 раз.

При использовании массива ключей, который занял дополнительные 19% памяти, время сортировки пузырьком сократилось до 1010241 микросекунд, то есть выигрыш во времени равен 379%, а время быстрой сортировки уменьшилось до 639 микросекунд, и для неё выигрыш во времени, соответственно, составил 17%.

После выбора оптимального алгоритма сортировки и использования дополнительного массив ключей, скорость работы уменьшилась до 7580 раз.

**Контрольные вопросы:**

1. Память выделяется таким образом, чтобы вместить наибольшее из возможных полей

2. Если в программе на языке Си в вариантную часть ввести данные, несоответствующие описанным, дальнейшее поведение будет считаться неопределённым

3. За обеспечение доступа к одному и тому же полю объединения ответственность несёт программист

4. Таблица ключей представляет собой характеристику основной таблицы по одному или нескольким полям. Ключ состоит из идентификатора (например, порядкового номера) записи в основной таблицы и копии одного или нескольких полей, по которым осуществляется сортировка записей.

5. Если запись содержит большое количество полей, а сортировка производится лишь по нескольким из них, то эффективнее будет использовать массив ключей. Если, например, запись состоит из малого количества полей, то сортировка исходной таблицы без использования массива ключей займёт меньше времени

6. Если важно минимизировать используемую память, используются алгоритмы с константной сложностью по памяти, например сортировка Шелла. Если в приоритете стоит быстродействие, то стоит рассмотреть сортировку слиянием, которая требует дополнительную память, но работает быстрее, чем вышеупомянутая сортировка Шелла. В зависимости от назначения таблиц, возможно использование неустойчивых сортировок, которые дают дополнительный выигрыш по времени (например, быстрая сортировка) или по памяти (например, пирамидальная сортировка).