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 №2 ПО ДИСЦИПЛИНЕ: ТИПЫ И СТРУКТУРЫ ДАННЫХ

Записи с вариантами. Обработка таблиц

Вариант 1

Студент **Абдуллаев Ш. В.**

Группа **ИУ7-34Б**

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

Студент _____ **Абдуллаев Ш. В.**

Преподаватель _____ **Силантьева А. В.**

2024

Описание условия задачи

Создать таблицу, содержащую не менее 40-ка записей. Упорядочить данные в ней по возрастанию ключей, двумя алгоритмами сортировки, где ключ – любое невариантное поле используя: а) саму таблицу, б) массив ключей. Возможность добавления и удаления записей в ручном режиме, просмотр таблицы, просмотр таблицы в порядке расположения таблицы ключей обязательна. Осуществить поиск информации по варианту. Ввести список литературы, содержащий фамилию автора, название книги, издательство, количество страниц, вид литературы:

1. Техническая:
    a. Отрасль
    b. Отечественная / переводная
    c. Год издания
2. Художественная:
    a. Тип литературы (роман, пьеса, поэзия)
3. Детская
    a. Минимальный возраст
    b. Тип детской литературы (стихи, сказки)

Вывести список всех романов указанного автора.

Описание исходных данных

Исходные данные представляют собой информацию о книга определенного вида. Каждая запись содержит следующие атрибуты: фамилию автора, название книги, издательство, количество страниц, вид литературы. В свою очередь вид литературы может быть техническая, содержащая отрасль, национализированность, год издания, художественная, содержащая тип литературы (роман, пьеса, поэзия), а также детская, содержащая минимальный возраст и тип детской литературы (стихи, сказки).

Имеются следующие ограничения:

1. Ограничение на количество записей в файле: программа обрабатывает корректный файл, содержащий не более 1000 записей. Если файл содержит больше записей, программа завершает работу с ошибкой.
2. Ограничение на формат данных: программа требует соблюдения строгого формата ввода данных. Например, для возраста и количества страниц должны использоваться только числовые значения, а для типа литературы — только определённые допустимые значения.
3. Ограничение на длину строк для фамилии автора, названия книги, издательства и отрасли: максимальная длина фамилии автора, названия книги, издательства и отрасли составляет 15 символов. Если длина превышает это значение, программа завершает работу с ошибкой.
4. Ограничение на количество страниц: количество страниц книги должно находиться в диапазоне от 1 до 1800. В случае выхода за эти пределы программа завершает работу с ошибкой.
5. Ограничение на типы литературы: программа обрабатывает только следующие типы литературы: техническая, художественная, детская. Другие типа не обрабатываются программой.
6. Ограничение на национальность: национальность автора должна быть выбрана только из двух допустимых значений (отечественная / переводная). Другие значения не обрабатываются программой.
7. Ограничение на год издания: год издания книги должен находиться в пределах от 1455 до 2024. Ввод значения за пределами этого диапазона приводит к ошибке.
8. Ограничение на типы художественной и детской литературы: программа поддерживает только следующие подтипы литературы: художественная литература (роман, пьеса, поэзия), детская литература (стихи, сказки). Другие типа не обрабатываются программой.
9. Ограничение на минимальный возраст читателя: минимальный возраст читателя должен находиться в диапазоне от 0 до 112 лет. Если возраст выходит за эти пределы, программа завершает работу с ошибкой.

Допустимый ввод	**Недопустимый ввод**
Введите фамилию автора: Artur	Введите фамилию автора: Aa…a
Введите количество страниц: 122	Введите количество страниц: hello
Введите год издания: 1999	Введите год издания: 3000
Введите минимальный возраст: 3	Введите минимальный возраст: 300

Таблица 1. Примеры ввода

Описание результатов

0. Выход из программы: пользователь из меню может выйти из программы.
1. Ввод данных: пользователь может ввести данные о книгах, указывая все вышеперечисленные атрибуты.
2. Вывод данных: пользователь может выбрать пункт меню для просмотра данных, которые были введены.
3. Вывод таблицы ключей: программа предоставляет возможность вывода таблицы ключей, основанной на кол. страниц.
4. Добавление записи: пользователь может добавлять новые записи о книге.
5. Удаление записи: программа позволяет удалять записи о книге.
6. Сортировка таблицы: предоставляются два варианта сортировки данных по кол. страниц - быстрая и медленная.
7. Сортировка массива ключей: программа также предоставляет возможность быстрой и медленной сортировки массива ключей.
8. Вывод данных по таблице ключей: пользователь может просматривать данные, основанные на таблице ключей.
9. Вывод таблицы эффективности: программа выводит таблицу, отображающую время выполнения различных операций.
10. Вывод списка всех романов указанного автора: по запросу пользователь может получить список всех романов по введенному автору.

Описание задачи, реализуемой программой

Цель работы программы заключается в создании инструмента, который позволяет управлять информацией о различной литературе. Это включает в себя ввод, хранение, обработку и вывод данных.

Способ обращения к программе

Обращения к программе пользователем происходит с помощью вызова исполняемого файла (app.exe).

Описание возможных аварийных ситуаций и ошибок пользователя

1. Не передан датасет. Программа вернет NO_DATA_ERROR.
2. Неудачный ввод команды. Программа вернет INPUT_COMMAND_ERR.
3. Ошибка открытия файла для чтения данных. Программа вернет OPEN_FILE_ERROR.
4. Достигнуто максимальное количество записей. Программа вернет OVERFLOW_BOOKS_ERROR.
5. Передан пустой файл. Программа вернет OPEN_FILE_ERROR.
6. Ошибка при чтении номера книги. Программа вернет READ_NUM_BOOK_ERR.
7. Некорректный номер книги. Программа вернет INVALID_NUM_BOOK_ERR.
8. Фамилия автора большей длины. Программа вернет OVERFLOW_LASTNAME_ERR.
9. Имя автора большей длины. Программа вернет OVERFLOW_NAME_ERR.
10. Издатель имеет большую длину. Программа вернет OVERFLOW_PUBLISHER_ERR.
11. Ошибка при чтении количества страниц. Программа вернет READ_PAGES_ERR.
12. Некорректное количество страниц. Программа вернет INVALID_CNT_PAGES_ERR.

13. Ошибка при чтении типа книги. Программа вернет READ_TYPE_BOOK_ERR.
14. Некорректный тип книги. Программа вернет INVALID_TYPE_BOOK_ERR.
15. Название отросли превышает допустимую длину. Программа вернет OVERFLOW_BRANCH_ERR.
16. Ошибка при чтении национализированности. Программа вернет READ_NATIONAL_ERR.
17. Некорректная национализированность. Программа вернет INVALID_NATIONAL_ERR.
18. Ошибка при чтении года. Программа вернет READ_YEAR_ERR.
19. Некорректный год. Программа вернет INVALID_YEAR_ERR.
20. Ошибка при чтении типа худ. литературы. Программа вернет READ_ART_TYPE_ERR.
21. Некорректный тип худ. литературы. Программа вернет INVALID_ART_TYPE_ERR.
22. Ошибка при чтении минимального возрастаю. Программа вернет READ_MIN_AGE_ERR.
23. Некорректный минимальный возраст. Программа вернет INVALID_MIN_AGE_ERR.
24. Ошибка при чтении типа детской литературы. Программа вернет READ_CHILD_TYPE_ERR.
25. Некорректный тип детской литературы. Программа вернет INVALID_CHILD_TYPE_ERR.
26. Ошибка считывания поля для удаления записи. Программа вернет READ_DEL_FIELD_ERR.
27. Некорректное поле для удаления записи. Программа вернет INVALID_DEL_ERR.

Описание внутренних структур данных

Программа содержит в себе структуру, которая используется для хранения информации о книге, которая представлена в Листинге 1.

```
#define LASTNAME_LEN 15
#define NAME_LEN 15
#define PUBLISHER_LEN 15
#define BRANCH_LEN 15

typedef enum
{
   technical,
   artistic,
   childish
} type_t;

typedef enum
{
   novel,
   play,
   poetry
} art_type_t;

typedef enum
{
   poems,
   fairytales
} child_type_t;

typedef struct
```

```
{
   int ind;
   char lastname[LASTNAME_LEN + 1];
   char name[NAME_LEN + 1];
   char publisher[PUBLISHER_LEN + 1];
   int page_cnt;
   type_t type;
   union
   {
      struct
      {
         char branch[BRANCH_LEN + 1];
         int isDomestic;
         int year;
      } technical;

      struct
      {
         art_type_t type;
      } artistic;

      struct
      {
         int min_age;
         child_type_t type;
      } childish;
   } details;
} book_t;
```

Листинг 1. Структура book_t

Рассмотрим каждое поле структуры:

- ind: целочисленное значение, указывающее номер (идентификатор) книги в системе.
- lastname: строка, содержащая фамилию автора книги, длиной до 15 символов.
- name: строка, содержащая имя автора книги, длиной до 15 символов.
- publisher: строка, содержащая название издательства книги, длиной до 15 символов.
- page_cnt: целочисленное значение, указывающее количество страниц в книге.
- type: тип литературы, задающее один из трех возможных вариантов: техническая, художественная, детская литература.
- details: объединение, содержащее дополнительные данные в зависимости от значения поля type. Если:
    - type равен technical:
        - branch: строка, представляющая отрасль науки или техники, длиной до 15 символов.
        - isDomestic: целочисленное значение, указывающее, является ли книга отечественной (например, 1 — отечественная, 0 — переводная).
    - type равен artistic:
        - type: жанр художественной литературы:
            - novel — роман,
            - play — пьеса,
            - poetry — поэзия.
    - type равен childish:
        - min_age: минимальный возраст, с которого рекомендуется читать книгу.
        - type: тип детской литературы:
            - poems — стихи,
            - fairytales — сказки.

Таким образом, структура book_t описывает книгу с общими сведениями (фамилия, имя автора, издательство, количество страниц) и специфическими полями, зависящими от типа литературы.

Также программа содержит в себе структуру book_key_t, которая представляет собой сокращенную версию информации о книге в виде ключа – значения, которая представлена в Листинге 2.

```
typedef struct
{
   int ind;
   int page_cnt;
} book_key_t;
```

Листинг 2. Структура book_key_t

Рассмотрим каждое поле структуры:

- ind: целочисленное значение, указывающее номер (идентификатор) книги в системе.
- page_cnt: целочисленное значение, указывающее количество страниц в книге.

Описание алгоритма

1. Инициализация системы и выделение памяти.
2. Основной цикл программы начинается. Пользователю предоставляется меню с различными опциями, и программа ожидает ввода выбора пользователя.
3. В зависимости от выбора пользователя, программа выполняет следующие действия:
   - Выйти из программы.
   - Прочитать данные из файла.
   - Вывести данные.
   - Вывести таблицу ключей.
   - Добавить новую запись.
   - Удалить существующую.

- Отсортировать данные по кол. страниц с использованием быстрой сортировки.
- Отсортировать данные по кол. страниц с использованием медленной сортировки.
- Отсортировать таблицу ключей с использованием быстрой сортировки.
- Отсортировать таблицу ключей с использованием медленной сортировки.
- Вывести данные, используя таблицу ключей.
- Измерить время выполнения различных сортировок для данных и таблицы ключей, а также оценить использование оперативной памяти.
- Вывести список всех романов указанного автора.

4. После выполнения каждой операции программа возвращает пользователя в главное меню, где он может выбрать следующее действие.

Набор тестов

Описание	**Результат**
Чтение данных из файла	Успешное считывание данных из файла
Вывод данных	Успешный вывод данных на экран
Добавление записи	Успешное добавление новой записи
Сортировка	Успешная сортировка данных
Удаление записи	Успешное удаление выбранной записи
Сортировка таблицы ключей	Успешная сортировка таблицы ключей
Вывод данных по таблице ключей	Успешный вывод данных о книгах по таблице ключей

Таблица 2. Набор позитивных тестов

Описание	**Результат**
Попытка не передать файл	Возврат ошибки NO_DATA_ERROR

Попытка передать несуществующий файл	Возврат ошибки OPEN_FILE_ERROR
Попытка передать пустой файл	Возврат ошибки OPEN_FILE_ERROR
Попытка добавления записи с недопустимой длиной фамилии автора	Возврат ошибки LASTNAME
Попытка передать не валидное количество страниц	Возврат ошибки INVALID_CNT_PAGES_ERR
Попытка ввода символов вместо года выпуска	Возврат ошибки READ_YEAR_ERR
Попытка ввода не валидного минимального возраста	Возврат ошибки INVALID_MIN_AGE_ERR
Попытка ввести несуществующий тип детской литературы	Возврат ошибки INVALID_CHILD_TYPE_ERR

Таблица 3. Набор негативных тестов

Оценка эффективности

При запуске программы N = 500 раз, были получены следующие данные:

Количество записей, шт.	BubbleSort, нс		QuickSort, нс	
	Таблица	Массив ключей	Таблица	Массив ключей
10	178	174	394	285
50	4107	4126	4768	3684
100	17821	14782	14416	12374
500	368493	332831	191981	131517
1000	1527565	1313458	288372	231779

Таблица 4. Время сортировок

Объем занимаемой памяти массива ключей от все таблицы равна ~10%.

Количество записей, шт.	% роста скорости сортировки массива	% роста скорости сортировки массива

	ключей по сравнению с таблицей (BubbleSort)	ключей по сравнению с таблицей (QuickSort)
10	~2%	~38%
50	~1%	~29%
100	~21%	~17%
500	~11%	~46%
1000	~16%	~24%

Таблица 5. Сравнение эффективности сортировок

Выводы

В ходе выполнения лабораторной работы была разработана программа для управления данными о книгах определенного вида. Программа предоставляет пользователю возможность считывать данные из файла, добавлять новые записи, удалять существующие, а также проводить сортировку данных и анализировать их эффективность. Были реализованы основные функции, такие как считывание и вывод данных, добавление и удаление записей, а также сортировка данных с использованием различных алгоритмов. Программа также поддерживает работу с таблицей ключей, что позволяет ускорить поиск и сортировку данных.

Различные сортировки показали разную эффективность. Использование таблицы ключей оказалось эффективным способом ускорения сортировки и поиска данных. Оценка использования оперативной памяти показала, что программа расходует память в соответствии с размером данных и таблицей ключей, что позволяет контролировать объем используемой памяти.

Контрольные вопросы

1. Как выделяется память под вариантную часть записи?

Выделение памяти под вариантную часть записи происходит так: выделяется память для самого большого поля, входящего в состав этого объединения. Это означает, что память под вариантную часть выделяется однократно, и она может

быть использована для любого из полей объединения, но в каждый момент времени может хранить только одно из значений.

2. Что будет, если в вариантную часть ввести данные, несоответствующие описанным?

Если в вариативную часть введены данные, несоответствующие ожидаемым, это может привести к некорректной работе программы. Например, если ожидается ввод числа, а пользователь вводит текст, это может вызвать ошибку или некорректное поведение программы. Поэтому важно предусмотреть проверки и обработку некорректного ввода.

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

Ответственность за правильность выполнения операций с вариативной частью записи лежит на программисте. Программист должен уделять внимание правильному выделению и освобождению памяти, а также обработке данных в вариативной части, чтобы избежать утечек памяти и ошибок в программе.

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

Таблица ключей представляет собой структуру данных, которая используется для ускорения поиска, сортировки и доступа к данным в основной таблице. Она содержит ссылки (индексы, указатели) на записи в основной таблице. Таблица ключей полезна, когда требуется быстрый доступ к данным, отсортированным по определенному критерию.

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

Эффективность обработки данных в самой таблице или с использованием таблицы ключей зависит от конкретной задачи. Если часто выполняются операции поиска и сортировки данных, то использование таблицы ключей может значительно ускорить выполнение программы. Однако, если требуется

постоянный доступ и изменение данных, то работа с данными в основной таблице может быть более эффективной.

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

Выбор способа сортировки зависит от размера данных и требований к производительности. Быстрая сортировка, например, может быть предпочтительнее для больших объемов данных, так как она имеет лучшую временную сложность. Однако, она требует дополнительной памяти для стека вызовов. Медленная сортировка, такая как сортировка пузырьком, может быть менее эффективной, но более простой в реализации и не требует дополнительной памяти. Выбор зависит от конкретной задачи и компромиссов между временем выполнения и потребляемой памятью.