|  |  |
| --- | --- |
| **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**

КАФЕДРА **ПРОГРАММНАЯ ИНЖЕНЕРИЯ (ИУ7)**

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.03.04** Программная инженерия

**Отчет**

|  |  |
| --- | --- |
| **По лабораторной работе №** | 2 |

**Название:**

Записи с вариантами, обработка таблиц

**Дисциплина:** Типы и структуры данных

|  |  |  |  |
| --- | --- | --- | --- |
| Студент | ИУ7-31Б |  | К. В. Морозов |
|  | (Группа) |  | (И.О. Фамилия) |
| Преподаватель |  |  |  |

Москва, 2022

**Описание условия задачи**

Спроектировать структуру позволяющую обрабатывать следующие данные: список литературы, содержащий фамилию автора, название книги, издательство, количество страниц, вид литературы (1: техническая – отрасль, отечественная,переводная, год издания; 2: художественная – роман, пьеса,стихи; 3: детская – сказки, стихи). Вывести списокотечественной технической литературы по указанной отраслиуказанного года.

**Описание ТЗ**

1. **Описание исходных данных**

Исходными данными является структурированная в файле информация об экземплярах книг.

Каждая запись содержит:

1. Фамилию автора книги
2. Название книги
3. Издательство
4. Количество страниц
5. Вид литературы (1, 2, 3)
6. (1: Техническая) – (отрасль, отечественная или переводная, год издания)
7. (2: Художественная) – (роман, пьеса,стихи)
8. (3: Детская) – (сказки, стихи)

Примечание: от выбранного вида литературы (1, 2, 3) зависят следующие параметры книги.

Данные программа может получать 2-мя способами: загрузка из заранее заготовленного файла “tables.txt” или методом добавления записей в конец внутри самой программы (с помощью соответствующего пункта меню (см. ниже)).

Программа работает в виде повторяющегося меню, так что при работе с программой пользователь на каждой итерации выбирает соответствующий пункт меню и, в зависимости от него, возможно вводит требуемые данные. При некорректных входных данных пользователь сможет дальше пользоваться программой, пока сам из неё не выйдет с помощью соответствующего пункта меню.

**Ограничения и допущения:**

- Наименования не должны превышать 30 символов

- Наименования не должны содержать в себе символ “;”

- Все численные поля должны быть только целочисленные

- Поле года издания не может быть отрицательным или нулевым или быть более 2022

- Поле количества страниц не может быть отрицательным или нулевым или быть более 99999

- Наименования и цифры не должны быть пустыми

- Наименования и цифры не должны начинаться с пробела

- Хранение вида литературы в данных является целочисленным

- Использование символов не из ASCII не позволяет программе работать корректно

- Максимальное количество записей в структуре – 999

- При повторных загрузках данных из файла, таблицы будут перезаписаны

- Все параметры в файле должны быть разделены “;”. В конце строки также “;”

- После пункта анализа, таблица ключей и сама таблицы сбрасываются к состоянию загрузки из файла

1. **Описание результатов программы**

1. Таблицы ключей

2. Таблица записей

3. Результат выборки

4. Вывод таблицы по ключам

5. Результаты замеров эффективности

**Описание задачи, реализуемой в программе**

Программа имеет интерфейс в виде меню, который позволяет совершать следующие действия:

0. Выход из программы

1. Загрузка таблицы из файла

2. Добавление записи в конец таблицы

3. Удаление записи таблицы по значению количества страниц

4. Сортировка таблицы ключей

5. Сортировка исходной таблицы

6. Вывод сортировки исходной таблицы, используя отсортированную таблицу ключей

7. Вывод результатов сравнения эффективности работы программы при обработке данных в исходной таблице и в таблице ключей

8. Вывод таблицы и таблицы ключей

9. Поиск списка отечественной технической литературы по указанной отрасли указанного года

**Способ обращения к программе**

Обращение к программе происходит через командную строку, путём запуска файла app.exe.

**Описание возможных аварийных ситуаций и ошибок пользователя**

Т.к реализация программы выполнена в виде циклического меню, то все ошибки, совершенные пользователем не приведут к аварийной ситуации, а будут обработаны с выводом соответствующего сообщения.

Эти ошибки могут быть следующего рода:

* Некорректные данные в файле
* Файл не существует
* Некорректный ввод наименований
* Некорректный ввод чисел
* Некорректный ввод условий, для произведения действий
* Попытка взаимодействия с таблицей, при её отсутствии

**Описание внутренних структур данных**

|  |
| --- |
| // описание экземпляра технической литературы struct technical\_literature\_instance {  char sphere[SPHERE\_NAME + 1];  int domestic\_or\_translated;  int year\_of\_publication; };  // описание типа художественной литературы struct artistic\_literature {  int artistic\_type; };  // описание типа детской литературы struct children\_literature {  int children\_type; };  typedef struct technical\_literature\_instance technical\_literature\_instance; typedef struct artistic\_literature artistic\_literature; typedef struct children\_literature children\_literature;  // описание вида литературы union literature\_type {  technical\_literature\_instance technical\_type;  artistic\_literature artistic\_type;  children\_literature children\_type; };  typedef union literature\_type literature\_type;  // описание экземпляра литературы (элемент таблицы) struct literature\_instance {  char author\_surname[AUTHOR\_SURNAME + 1];  char book\_title[BOOK\_TITLE + 1];  char publisher\_name[PUBLISHER\_NAME + 1];  int number\_of\_pages;  int int\_type;  literature\_type type; };  typedef struct literature\_instance literature\_instance;  // описание типа ключа (значение - индекс) struct key\_struct {  int key\_value;  int key\_index; };  typedef struct key\_struct key\_struct;  // описание целой таблицы struct table {  literature\_instance literatures\_instances[MAX\_FIELDS];  key\_struct key\_instances[MAX\_FIELDS];  int fields\_count; };  typedef struct table table; |

Пояснения: в программе существует 3 вариантных поля (каждое представляет из себя отдельную структуру, которые объеденены в union literature\_type (тип литературы)). Данное объеденение также само является частью структуры, которая хранит информацию об одном экземпляре книги. Помимо этого существует и структура экземпляра ключа, которая хранит индекс поля ключа и его значение. Все они используются в главной структуре в качестве массива экземпляров книг и ключей.

Размеры:

MAX\_FIELDS = 999

AUTHOR\_SURNAME = BOOK\_TITLE = PUBLISHER\_NAME = SPHERE\_NAME = 30

**Тесты**

**Положительные тесты**

1. Успешная сортировка таблицы
2. Успешная сортировка таблицы ключей
3. Успешный вывод таблицы по отсортированной таблице ключей
4. Успешная загрузка таблицы в программу из файла
5. Успешное добавление записи таблицы в конец
6. Успешное удаление записи таблицы по значению количества страниц
7. Успешный поиск отечественной технической литературы по полям

**Негативные тесты**

1. Файл отсутствует
2. Файл пустой
3. Файл заполнен не в соответствии с требованиями
4. Неверный выбор пункта меню
5. Неверно задано число записей таблицы в файле
6. Число записей таблицы в файле меньше, либо равно 0
7. Число полей таблицы превышает максимально установленное значение
8. В файле записаны некорректные лексически поля записей
9. В файле записаны некорректные логически поля значений
10. Пользователь вводит некорректное лексически наименование
11. Пользователь вводит некорректное лексически число
12. В файле записаны некорректные по длине поля записи
13. Пользователь вводит некорректные по длине поля записей
14. Попытка сортировки пустой таблицы
15. Попытка удаления в пустой таблице
16. Попытка просмотра при пустой таблице
17. Попытка сравнения результатов при пустой таблице
18. Попытка удаления элементов, которых нет

**Сравнение сортировок**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Размер | Быстра сортировка (таблица ключей) | Медленная сортировка (таблица ключей) | Быстрая сортировка (таблица) | Медленная сортировка (таблица) |
| 99 | 2 мкс | 14 мкс | 3 мкс | 50 мкс |
| 499 | 14 мкс | 342 мкс | 19 мкс | 1197 мкс |
| 999 | 33 мкс | 1341 мкс | 43 мкс | 4804 мкс |

Примечание: мкс = с \* 10 ^ -6

В качестве быстрой сортировки O(n \* log(n)) использовался quick\_sort.

В качестве медленной сортировки O(n ^ 2) использовался bubble\_sort.

Таблица, насколько в процентах медленнее bubble\_sort

|  |  |  |
| --- | --- | --- |
| Размер | Таблица ключей | Таблица |
| 99 | 85.7% | 94.0% |
| 499 | 95,9% | 98.4% |
| 999 | 97.5% | 99.1% |

Можно заметить, что при увеличении размера таблицы, разницы между quick\_sort и bubble\_sort значительно возрастает. Также, время сортировки таблицы гораздо медленнее, чем таблицы ключей.

**Затраты по памяти**

|  |  |  |
| --- | --- | --- |
| Размер | Таблица ключей | Таблица |
| 99 | 792 б | 14256 б |
| 499 | 3992 б | 71856 б |
| 999 | 7992 б | 143856 б |

Примечание: б = байт

Таблица, насколько в процентах идёт проигрыш по памяти таблицы, если использовать дополнительно таблицу ключей.

|  |  |
| --- | --- |
| Размер | Проигрыш |
| 99 | 5.56% |
| 499 | 5.56% |
| 999 | 5.56% |

Следовательно, проигрыш по памяти при использовании таблицы ключей составляет приблизительно 5.56%.

**Вывод**

Чем больше записей содержится в таблице, тем более большой выигрыш во времени у быстрых сортировок перед медленными. Также при работе с большими объёмами данных, гораздо более рентабельно по скорости (при малом проигрыше в памяти) использовать таблицу ключей. В дополнение к этому, при работе с несколькими типами полей, часто выгодно использовать объединения.

**Ответы на контрольные вопросы**

1.Как выделяется память под вариативную часть записи?

Вариативная часть записи хранит в одном участке памяти все вариативные поля. Память, выделенная под вариантную часть, будет зависеть от максимального по размеру поля.

2.Что будет, если в вариативную часть ввести данные, несоответствующие с описанными?

Будет некорректное отображение данных.

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

За правильностью выполнения операций с вариативной частью записи должен следить программист.

4.Что представляет из себя таблица ключей и зачем она нужна?

Таблица ключей представляет из себя массив экземпляров такой структуры, что хранит в себе значение какого-либо ключа и его индекс из обычной таблицы.

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

Если размер таблицы относительно большой, то для ускорения работы рекомендуется обрабатывать таблицу ключей.

Если размер таблицы относительно невелик, то нет особого смысла в выделении дополнительной памяти под таблицу ключей.

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

Предпочтительнее использовать устойчивые сортировки, т.к они позволяют не совершать лишних перестановок. Более того, лучше всего использовать сортировки, которые имеют ассимптотическую сложность O(n \* log(N))