|  |  |
| --- | --- |
| 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 \_Программное обеспечение ЭВМ и информационные технологии\_\_\_\_

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №2**

**«ЗАПИСИ С ВАРИАНТАМИ, ОБРАБОТКА ТАБЛИЦ»**

Студент\_\_\_\_\_\_Чыонг Ван Хао\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*фамилия, имя, отчество*

Группа\_\_\_\_\_*ИУ7И-31Б\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

*Вариант 24*

Принял : Силантьева А. В.

**Цель работы:** приобрести навыки работы с типом данных «запись» («структура») содержащим вариантную часть, и с данными, хранящимися в таблицах. Оценить относительную эффективность программы (в процентах) по времени и по используемому объему памяти в зависимости от используемого алгоритма и от объема сортируемой информации.

**Описание 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 – возраст, используя: а) саму таблицу, б) массив ключей. (Возможность добавления и удаления записей в ручном режиме обязательна). Осуществить поиск информации по варианту.

Имеются описания:

Type жилье = (дом, общежитие);

Данные:

Фамилия, имя, группа, пол (м, ж), возраст, средний балл за сессию, дата поступления

адрес:

дом: (улица, №дома, №кв);

общежитие: (№общ, №комн.);

Ввести общий список студентов.

Вывести список студентов, указанного года поступления, живущих в общежитии .

**Техническое задание**

**Входные данные**

Программа для получения данных из входного файла.

Menu :

1 - Add record to table

2 - Delete record from table

3 - Print Table

4 - Print Key Table

5 - Print Table by Key

6 - Sort Table

7 - Sort Key Table

8 - Find student

9 - Show comparations of sort(time)

0 – Exit

1. Ввод данных в таблицу

1 - Add record from file ?

2 - Add record by input ?

0 - Go to Menu

* Импортировать данные из файла
* Ввести данные вручную
* Вернуться в главное меню

1. Удалить запись по индексу
2. Отобразить таблицу
3. Отобразить таблицу ключей
4. Отобразить таблицу по ключу
5. Отсортировать таблицу данных
6. Отсортировать таблицу ключей
7. Найти интересующих студентов-ввод-год поступления .
8. Поэкспериментируйте с функциями сортировки

**Выходные данные**

Программа может вывести: неотсортированную таблицу с данными из файла, отсортированную таблицу с данными из файла, таблицу ключей, отсортированную таблицу ключей, отобразить интересующих студентов (живущих в общежитии указанного года поступления)число: строка, которая может состоять только из символов, время операций сортировки и объем используемой памяти.

**Функция программы**

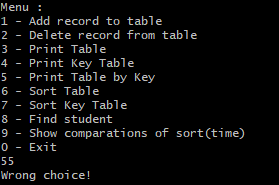
Программа получает данные в виде записей и сортирует их.

**Обращение к программе**

Программа запускается из терминала командой «./app.exe» или «make app» в директории с программой.

**Возможные аварийные ситуации и ошибки пользователя**

1. Ввести неправильный выбор.



На выходе сообщение: «Wrong choice!»

2. Попробуйте манипулировать пустой таблицей.

На выходе сообщение: «Table is empty now!» или «TABLE EMPTY! »

3. Введите неправильное расположение записи, которую хотите выбрать.

На выходе сообщение: «Error input!»

4. Файл не найден

На выходе сообщение: «ERROR NO FILE! »

5. Пустой файл

На выходе сообщение: «ERROR EMPTY FILE!»

6. Ошибка при получении данных из файла

На выходе сообщение: «ERROR: EMPTY – RECORD ?? »

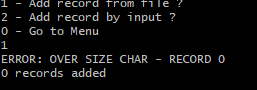
7. Числовые данные превышены



На выходе сообщение: «ERROR: WRONG DATA – RECORD ??»

8. Символ переполнения





На выходе сообщение: «ERROR: OVER SIZE CHAR - RECORD??»

9. Переполнение. количество записей

На выходе сообщение: « TABLE NOW IS FULL!»

10. Введены неверные данные поиска

На выходе сообщение: « ERROR - WRONG YEAR!» или « ERROR!»

**Внутренняя структура данных**

Для хранения записей в таблице использую структуры:

MAX\_CHAR\_SIZE = 15

typedef struct

{

    char street[MAX\_CHAR\_SIZE];

    int num\_house;

    int num\_apartment;

} houses;

typedef struct

{

    int num\_dorm;

    int num\_room;

} dorms;

typedef union

{

    houses house;

    dorms dorm;

} type\_lives;

typedef struct

{

    int type\_home;

    char family[MAX\_CHAR\_SIZE];

    char name[MAX\_CHAR\_SIZE];

    char group[MAX\_CHAR\_SIZE];

    int sex;

    int age;

    int mark;

    char date[MAX\_CHAR\_SIZE];

    int index;

    type\_lives home;

} student;

typedef struct

{

    int age;

    int index;

} keys;

**Функции программы**

int check\_file(FILE \*file);

Описание: выполнить проверку входного файла.

Входные значения: входной файл

Выходные значения: код ошибки и сообщение (если есть).

int read\_number(FILE \*file, int \*num, const int min, const int max, const int \*index);

Описание: выполнить получить целое число из ввода.

int read\_char(FILE \*file, char \*str, int size, const int \*index);

Описание: функция, выполняющая ввод строки данных.

void read\_from\_file(FILE \*file, student \*person, keys \*key, int \*index);

Описание: получить данные из входного файла.

void print\_table(student \*person, keys \*key, const int \*size, const int by\_key);

Описание: Отобразить таблицу.

void add\_record\_to\_table(student \*person, keys \*key, int \*index);

Описание: вручную добавить запись в таблицу.

void find\_student(student \*person,const int \*size);

Описание: Найти интересующих студентов.

void print\_key\_table(keys \*key, const int \*size);

Описание: Отобразить таблицу ключей.

void buble\_sort(student \*person, const int size);

Описание: Отсортировать таблицу данных.

void insert\_binary\_sort(student \*person, const int size);

Описание: Отсортировка таблицы данных методом вставки с бинарным поиском.

void buble\_sort\_key(keys \*key, const int size);

Описание: Отсортировать таблицу ключей.

void insert\_binary\_sort\_key(keys \*key, const int size);

Описание: Отсортировать таблицу ключей методом вставки с бинарным поиском.

void random\_table\_data(student \*person, keys \*key, const int size);

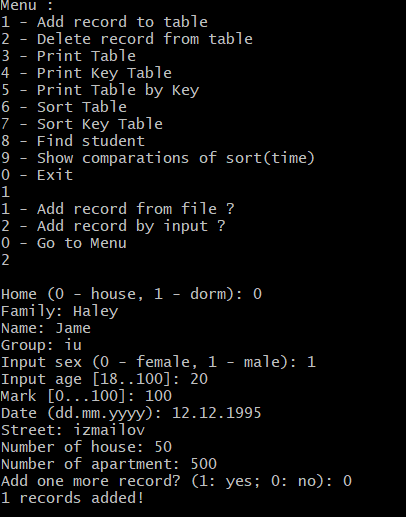
Описание: сгенерировать таблицу со случайными данными.

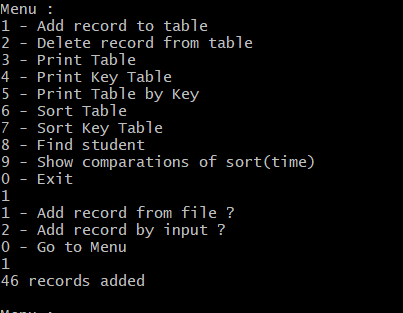
void test\_sort\_table();

Описание: сравнивать с функциями сортировки.

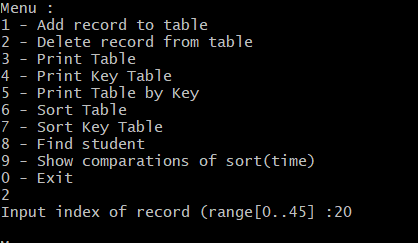
**Тесты**

Ввод значений

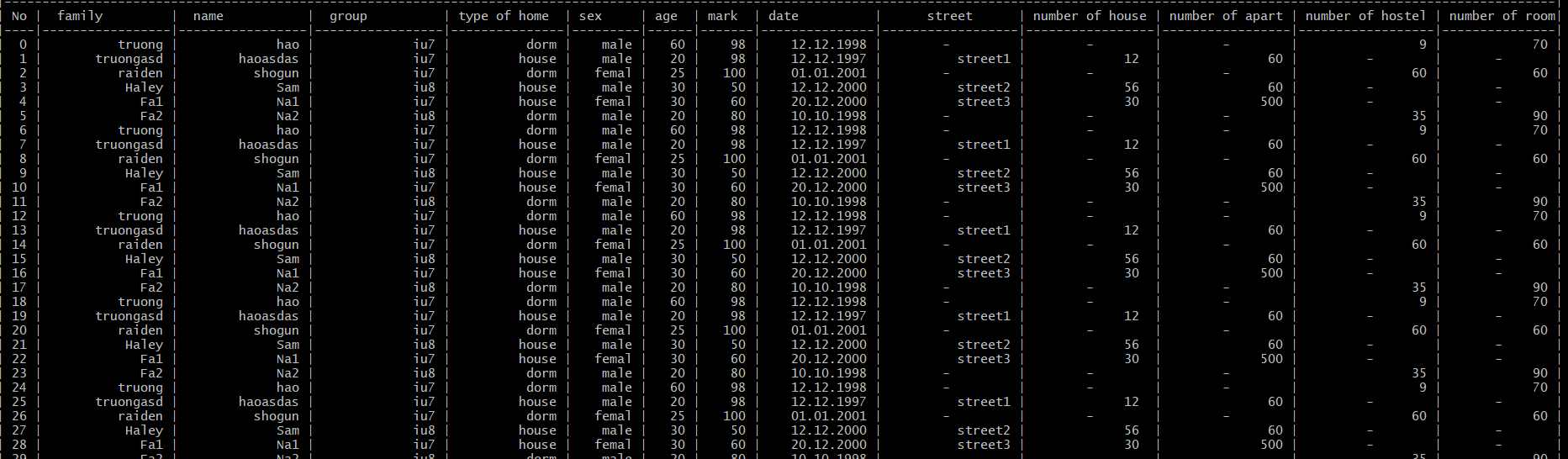




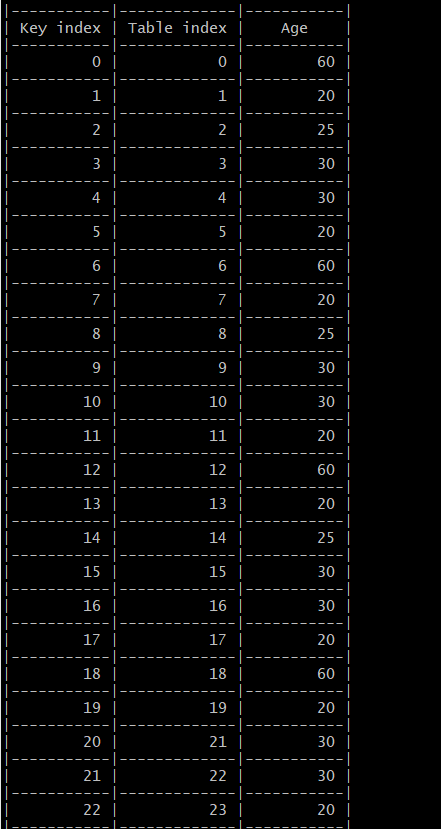
Удаление записи:



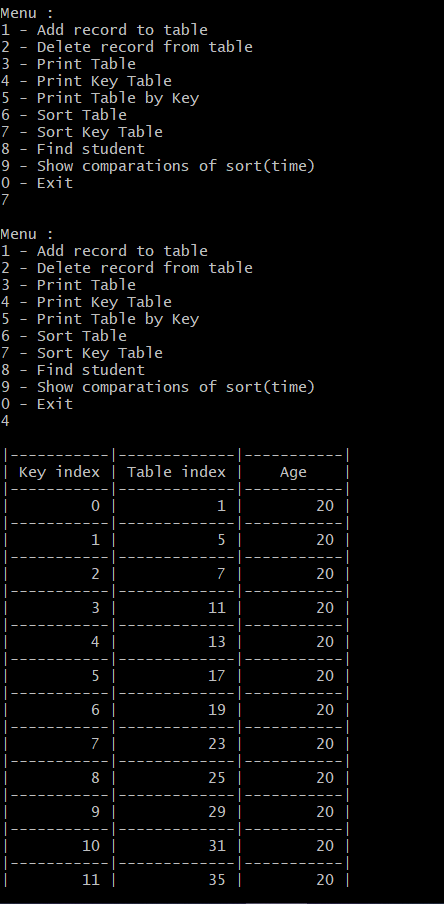
Отображение таблицы



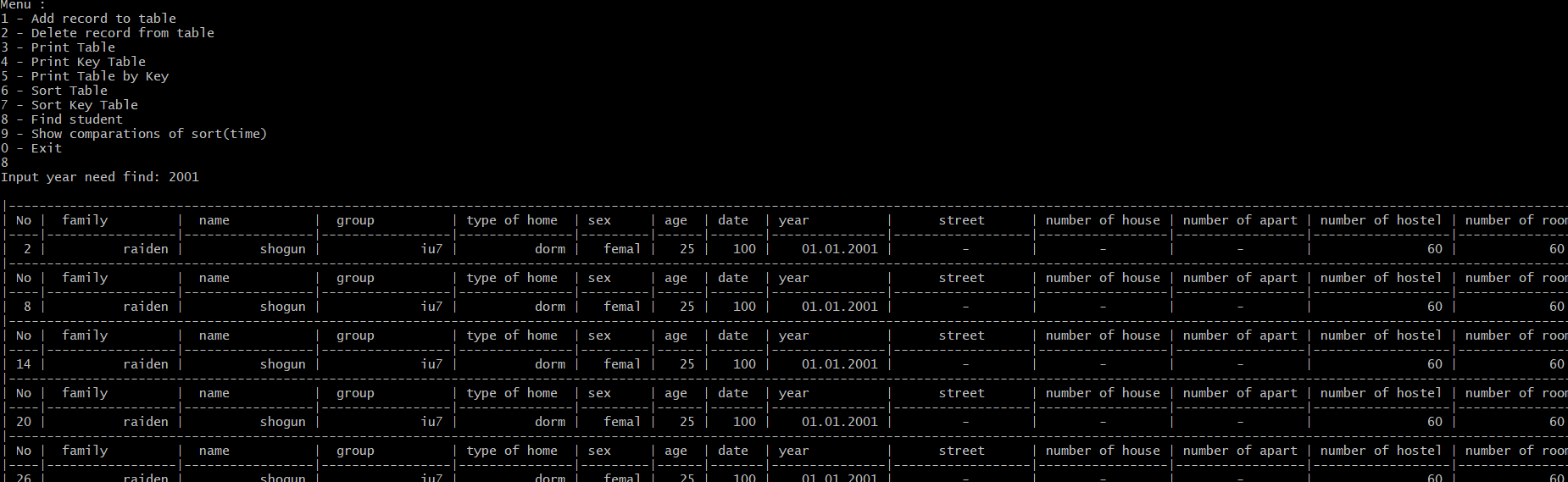
Вывод таблицы ключей на экран:



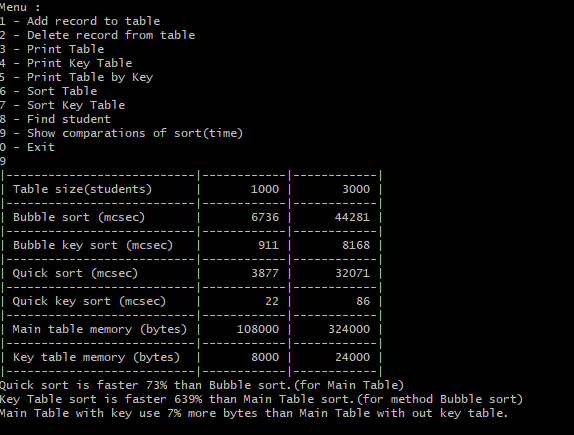
Вывод отсортированной таблицы ключей:



Поиск по фильтру:



Сравнивать с функциями сортировки



**Выводы**

В ходе тестирования было выявлено, что использование таблицы ключей намного эффективнее по времени, чем сортировка целой таблицы. Использование таблицы ключей не эффективно по памяти, но в данной ситуации выбор данного метода оправдан, так как сортировка производиться по числовым данным, а не по текстовым.

Эффективность по времени:

quick sort быстрее bubble sort в 0.73 раз

Сортировка таблицы ключей быстрее сортировки таблицы в 6.39 раз

Эффективность по памяти:

Работа с таблицей ключей занимает на 7% больше памяти, чем работа с исходной таблицей.

**Контрольные вопросы**

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

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

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

Невозможно корректно прочитать данные.

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

Разработчик.

**4. 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 основной таблице с данными по индексу элемента.

При больших размерах таблиц поиск данных и их сортировка может потребовать больших затрат времени. Можно уменьшить время обработки (сортировки) таблицы за счет введения таблицы ключей.

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

При использовании таблицы ключей экономится время , так как отсутствует перестановка больших полей. Однако таблица ключей занимает дополнительную память, поэтому мы не можем достичь эффективности по памяти. Поэтому не стоит использовать таблицу ключей, если ключом являются текстовые данные. Выбор данных из основной таблицы в порядке, определенной таблицей ключей, также замедляет программу.

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

При сортировке таблиц эффективнее использовать таблицу ключей. При этом метод вставками с бинарным поиском работает быстрее, чем сортировка пузырьком.