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
2) Удалить запись по индексу
3) Отобразить таблицу
4) Отобразить таблицу ключей
5) Отобразить таблицу по ключу
6) Отсортировать таблицу данных
7) Отсортировать таблицу ключей
8) Найти интересующих студентов-ввод-год поступления .
9) 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

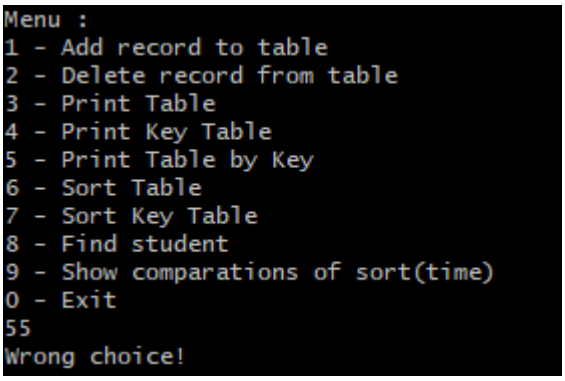
Программа получает данные в виде записей и сортирует их.

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! »

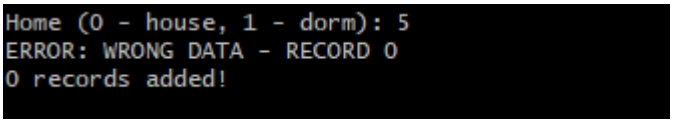
5. Пустой файл

На выходе сообщение: «ERROR EMPTY FILE!»

6. Ошибка при получении данных из файла

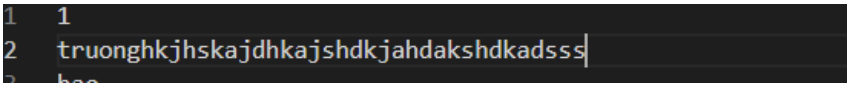
На выходе сообщение: «ERROR: EMPTY – RECORD ?? »

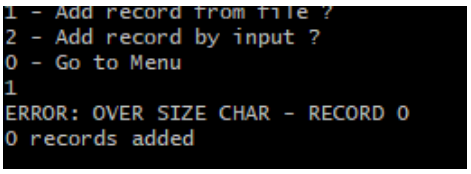
7. Числовые данные превышены



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

```
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
1 - Add record from file ?
2 - Add record by input ?
0 - Go to Menu
2

Home (0 - house, 1 - dorm): 0
Family: Haley
Name: Jame
Group: iu
Input sex (0 - female, 1 - male): 1
Input age [18..100]: 20
Mark [0...100]: 100
Date (dd.mm.yyyy): 12.12.1995
Street: izmailov
Number of house: 50
Number of apartment: 500
Add one more record? (1: yes; 0: no): 0
1 records added!
```

```
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
1 - Add record from file ?
2 - Add record by input ?
0 - Go to Menu
1
46 records added

Menu :
```

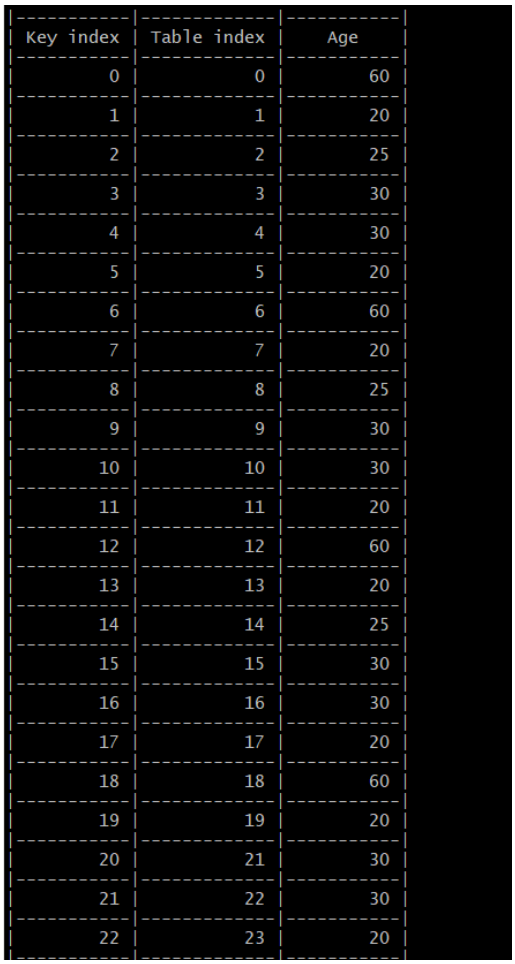
Удаление записи:

```
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
2
Input index of record (range[0..45] :20
```

Отображение таблицы

No	family	name	group	type of home	sex	age	mark	date	street	number of house	number of apart	number of hostel	number of room
0	truong	hao	iu7	dorm	male	60	98	12.12.1998	-	-	-	9	70
1	truongasd	haoasdas	iu7	house	male	20	98	12.12.1997	street1	12	60	-	-
2	raiden	shogun	iu7	dorm	femal	25	100	01.01.2001	-	-	-	60	60
3	Haley	Sam	iu8	house	male	30	50	12.12.2000	street2	56	60	-	-
4	Fa1	Na1	iu7	house	femal	30	60	20.12.2000	street3	30	500	-	-
5	Fa2	Na2	iu8	dorm	male	20	80	10.10.1998	-	-	-	35	90
6	truong	hao	iu7	dorm	male	60	98	12.12.1998	-	-	-	9	70
7	truongasd	haoasdas	iu7	house	male	20	98	12.12.1997	street1	12	60	-	-
8	raiden	shogun	iu7	dorm	femal	25	100	01.01.2001	-	-	-	60	60
9	Haley	Sam	iu8	house	male	30	50	12.12.2000	street2	56	60	-	-
10	Fa1	Na1	iu7	house	femal	30	60	20.12.2000	street3	30	500	-	-
11	Fa2	Na2	iu8	dorm	male	20	80	10.10.1998	-	-	-	35	90
12	truong	hao	iu7	dorm	male	60	98	12.12.1998	-	-	-	9	70
13	truongasd	haoasdas	iu7	house	male	20	98	12.12.1997	street1	12	60	-	-
14	raiden	shogun	iu7	dorm	femal	25	100	01.01.2001	-	-	-	60	60
15	Haley	Sam	iu8	house	male	30	50	12.12.2000	street2	56	60	-	-
16	Fa1	Na1	iu7	house	femal	30	60	20.12.2000	street3	30	500	-	-
17	Fa2	Na2	iu8	dorm	male	20	80	10.10.1998	-	-	-	35	90
18	truong	hao	iu7	dorm	male	60	98	12.12.1998	-	-	-	9	70
19	truongasd	haoasdas	iu7	house	male	20	98	12.12.1997	street1	12	60	-	-
20	raiden	shogun	iu7	dorm	femal	25	100	01.01.2001	-	-	-	60	60
21	Haley	Sam	iu8	house	male	30	50	12.12.2000	street2	56	60	-	-
22	Fa1	Na1	iu7	house	femal	30	60	20.12.2000	street3	30	500	-	-
23	Fa2	Na2	iu8	dorm	male	20	80	10.10.1998	-	-	-	35	90
24	truong	hao	iu7	dorm	male	60	98	12.12.1998	-	-	-	9	70
25	truongasd	haoasdas	iu7	house	male	20	98	12.12.1997	street1	12	60	-	-
26	raiden	shogun	iu7	dorm	femal	25	100	01.01.2001	-	-	-	60	60
27	Haley	Sam	iu8	house	male	30	50	12.12.2000	street2	56	60	-	-
28	Fa1	Na1	iu7	house	femal	30	60	20.12.2000	street3	30	500	-	-

Вывод таблицы ключей на экран:



Key index	Table index	Age
0	0	60
1	1	20
2	2	25
3	3	30
4	4	30
5	5	20
6	6	60
7	7	20
8	8	25
9	9	30
10	10	30
11	11	20
12	12	60
13	13	20
14	14	25
15	15	30
16	16	30
17	17	20
18	18	60
19	19	20
20	21	30
21	22	30
22	23	20

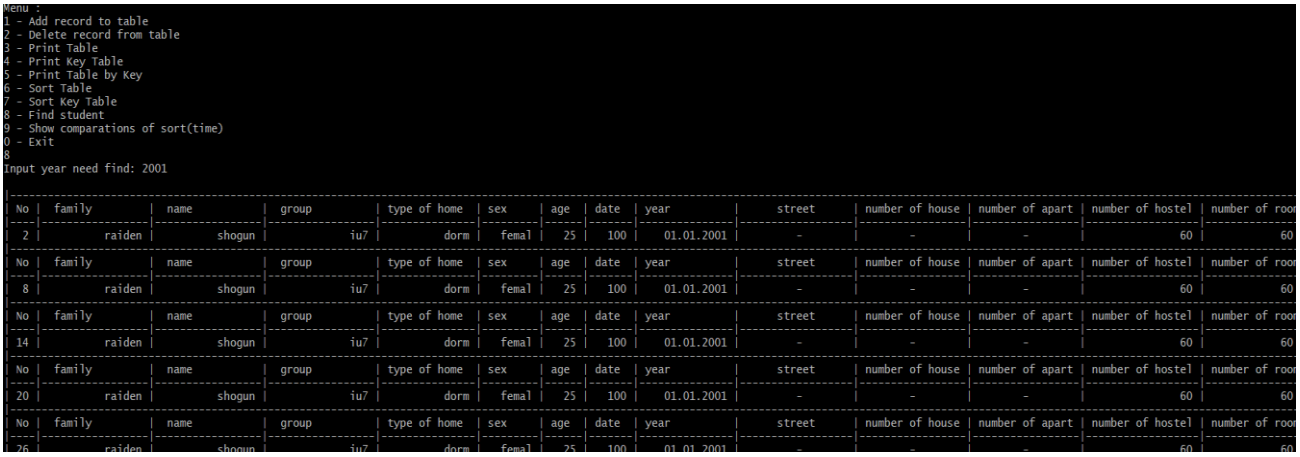
Вывод отсортированной таблицы ключей:

```
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
7

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
4

|-----------|-------------|-----------|
| Key index | Table index |    Age    |
|-----------|-------------|-----------|
|         0 |           1 |        20 |
|-----------|-------------|-----------|
|         1 |           5 |        20 |
|-----------|-------------|-----------|
|         2 |           7 |        20 |
|-----------|-------------|-----------|
|         3 |          11 |        20 |
|-----------|-------------|-----------|
|         4 |          13 |        20 |
|-----------|-------------|-----------|
|         5 |          17 |        20 |
|-----------|-------------|-----------|
|         6 |          19 |        20 |
|-----------|-------------|-----------|
|         7 |          23 |        20 |
|-----------|-------------|-----------|
|         8 |          25 |        20 |
|-----------|-------------|-----------|
|         9 |          29 |        20 |
|-----------|-------------|-----------|
|        10 |          31 |        20 |
|-----------|-------------|-----------|
|        11 |          35 |        20 |
```

Поиск по фильтру:



Сравнивать с функциями сортировки

```
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
9
|---------------------------|------------|------------|
| Table size(students)      |       1000 |       3000 |
|---------------------------|------------|------------|
| Bubble sort (mcsec)       |       6736 |      44281 |
|---------------------------|------------|------------|
| Bubble key sort (mcsec)   |        911 |       8168 |
|---------------------------|------------|------------|
| Quick sort (mcsec)        |       3877 |      32071 |
|---------------------------|------------|------------|
| Quick key sort (mcsec)    |         22 |         86 |
|---------------------------|------------|------------|
| Main table memory (bytes) |     108000 |     324000 |
|---------------------------|------------|------------|
| Key table memory (bytes)  |       8000 |      24000 |
|---------------------------|------------|------------|
Quick sort is faster 73% than Bubble sort.(for Main Table)
Key Table sort is faster 639% than Main Table sort.(for method Bubble sort)
Main Table with key use 7% more bytes than Main Table with out key table.
```

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.