Отчет по курсовой работе №9

по курсу: языки и методы программирования

студент группы : М8О-105Б-21 Козлов Егор Сергеевич, № по списку: 9

Адреса www, e-mail, jabber, skype: iamaghoulzxc@gmail.com

Работа выполнена: "29 мая 2022 г."

Преподаватель: каф. 806 В.К.Титов

Входной контроль знаний с оценкой: _____

Отчет сдан "___"_____20___г., итоговая оценка _____

Подпись преподавателя _____

1. Тема: Сортировка и поиск.

2. Цель работы: Составить и отладить программу на языке Си для сортировки таблицы заданным методом и бинарного поиска по ключу в таблице

3. Задание (*вариант 9*): Пирамидальная сортировка просеиванием. Вещественный ключ 4 байта

4. Оборудование(*лабораторное*):
ЭВМ _____, процессор _____, имя узла сети ____ с ОП ___ ГБ
НМД _____ ГБ. Терминал _____ адрес _____. Принтер _____
Другие устройства _____
Оборудование ПЭВМ студента, если использовалось:
Процессор Ryzen 7 5800 @ 8x 3.2 GHz , ОП 16384 МБ, НМД _____ГБ. Монитор Встроенный
Другие устройства _____

5. Программное обеспечение(*лабораторное*):
Операционная система семейства UNIX, наименование _____ версия _____
Интерпретатор команд: _____ версия _____
Система программирования: _____ версия _____
Редактор текстов: _____ версия _____
Утилиты операционной системы: _____
Прикладные системы и программы: _____
Местонахождение и имена файлов и программ данных: _____

Программное обеспечение ЭВМ студента, если использовалось:
Операционная система семейства UNIX, наименование Ubuntu версия 22.04
Интерпретатор команд: bash версия _____
Система программирования: C версия _____
Редактор текстов: Emacs версия _____
Утилиты операционной системы: _____
Прикладные системы и программы: _____
Местонахождение и имена файлов и программ данных: /usr/bin , а также /bin

6. Идея, метод, алгоритм решения задачи (в формах: словесной, псевдокода, графической [блок-схема, диаграмма, рисунок, таблица] или формальное описание с пред- и постусловиями)

Пирамидальная сортировка использует бинарное сортирующее дерево.

Сортировка начинается с построения *“пирамиды”* (бинарного дерева), которая содержит все элементы исходной последовательности. Максимальный элемент окажется в вершине дерева, поскольку все потомки этой вершины обязательно должны быть меньше. Затем, вершина пирамиды записывается в конец последовательности, а "пирамида"с удаленным максимальным элементом переформировывается. В результате, на её вершине оказывается максимальный из оставшихся элементов, он записывается на соответствующее место, и процедура продолжается до тех пор, пока "пирамида"не опустеет.

7. Сценарий выполнения работы [план работы, первоначальный текст программы в черновике (можно на отдельном листе) и тесты, либо соображения по тестированию].

kp9.cpp :

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

int sortFlag = 0, printFlag = 0;

struct row {
    float key;
    char *string;
};

int getSize(FILE *f) {
    int size = 0;
    char *buffer = new char[128];
    size_t l = sizeof(buffer);
    while (!feof(f)) {
        getline(&buffer, &l, f);
        size++;
    }
    delete[] buffer;
    rewind(f);
    return size;
}

float last;
void printTable(row *table, int size) {
    printf("------------------------------------------------------------\n");
    printf("| Key |                        Data                        |\n");
    printf("------------------------------------------------------------\n");
    for (int i = 0; i < size; ++i) {
        printf("|%f | %s", table[i].key, table[i].string);
        if ((last == table[i].key) && ((printFlag == 1))) printf("\n");
        if ((i == (size - 1)) && (printFlag == 0)) {
            last = table[i].key;
            printFlag = 1;
            printf("\n");
        }
        printf("------------------------------------------------------------\n");
    }
    printf("\n");
}

row *makeTable(FILE *keys, FILE *value, int size) {
    row *table = new row[size];
    for (int i = 0; i < size; ++i) {
        table[i].string = new char[128];
    }
    for (int i = 0; i < size; ++i) {
        fscanf(keys, "%f", &table[i].key);
        fgets(table[i].string, 128, value);
    }
    return table;
}

int binSearch(row *tab, float key, int size) {
    int l = 0, r = size - 1, mid;
    while (l <= r) {
        mid = (r + l) / 2;
        if (key == tab[mid].key) return mid;
        else if (key < tab[mid].key) r = mid;
```

```
        else if (key > tab[mid].key) l = mid + 1;
    }
    return -1;
}

void swapStrings(row *table, int a, int b) {
    row tmp;
    tmp = table[a];
    table[a] = table[b];
    table[b] = tmp;
}

void mixTable(row *tab, int size) {
    sortFlag = 0;
    int a, b;
    for (int i = 0; i < size; ++i) {
        a = rand() % size;
        b = rand() % size;
        swapStrings(tab, a, b);
    }
}

void reverseStrings(row *table, int size) {
    sortFlag = 0;
    for (int i = 0, j = size - 1; i < j; ++i, --j) swapStrings(table, i, j);
}

void sift(row *tab, int i, int bot) {
    int max;
    int flag = 0;
    while ((i * 2 <= bot) && (!flag)) {
        if (i * 2 == bot) max = i * 2;
        else if (tab[i * 2].key > tab[i * 2 + 1].key) max = i * 2;
        else max = i * 2 + 1;
        if (tab[i].key < tab[max].key) {
            swapStrings(tab, i, max);
            i = max;
        } else flag = 1;
    }
}

void heapSort(row *tab, int size) {
    for (int i = size / 2; i >= 0; --i) {
        sift(tab, i, size - 1);
    }
    for (int i = size - 1; i >= 1; --i) {
        swapStrings(tab, 0, i);
        sift(tab, 0, i - 1);
    }
}

void sortTable(row *tab, int size) {
    sortFlag = 1;
    heapSort(tab, size);
}

int main() {
    srand(time(0));
    FILE *firstFile = fopen("IN1", "r");
    if (firstFile == NULL) {
        printf("Can not open file 1\n");
        return 0;
    }
    FILE *secondFile = fopen("IN2", "r");
    if (secondFile == NULL) {
        printf("Can not open file 2\n");
        return 0;
    }
    int n = getSize(firstFile), action;
    row *tab = makeTable(firstFile, secondFile, n);
    fclose(firstFile);
    fclose(secondFile);
    printTable(tab, n);
    while (true) {
        printf("Menu\n"
               "1. Print table\n"
               "2. Mix table\n"
               "3. Reverse table\n"
               "4. Binary search\n"
```

```
                "5. Sort table\n"
                "0. Exit\n"
                "Choose an action =>");

        printf("Choose an action ===> ");
        scanf("%d", &action);

        switch (action) {
            case 1: {
                printTable(tab, n);
                break;
            }
            case 2: {
                if (sortFlag) {
                    printf("Enter the key: ");
                    float key;
                    scanf("%f", &key);
                    int search;
                    printf("\n");
                    search = binSearch(tab, key, n);
                    if (search == -1) {
                        printf("Element with such key is not found!\n");
                        break;
                    }
                    printf("Found the string:\n %s", tab[search].string);
                } else
                    printf("Table is not sorted!\n");
                break;
            }
            case 3: {
                sortTable(tab, n);
                break;
            }
            case 4: {
                mixTable(tab, n);
                break;
            }
            case 5: {
                reverseStrings(tab, n);
                break;
            }
            case 6:
                return 0;
        }
    }
}
```

Пункты 1-7 отчѐта составляются ***строго до*** *начала лабораторной работы.*

Допущен к выполнению работы. Подпись преподавателя _____

8. Распечатка протокола (подклеить листинг окончательного варианта программы с текстовыми примерами, подписанный преподавателем).

```
isitmuse@isitmuse:~/lab/secondSem/kp9$ cat head
--------------------------------
||||||КУРСОВАЯ  РАБОТА  №9||||||
|||||||СОРТИРОВКА И ПОИСК|||||||
|||ВЫПОЛНИЛ  СТУДЕНТ  ГРУППЫ||||
|||М8О-105Б-21  КОЗЛОВ  ЕГОР||||
\\\\\\\\\\\\\\\\||///////////////
isitmuse@isitmuse:~/lab/secondSem/kp9$ cat kp9.cpp
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

int sortFlag = 0, printFlag = 0;

struct row {
    float key;
    char *string;
};

int getSize(FILE *f) {
    int size = 0;
    char *buffer = new char[128];
    size_t l = sizeof(buffer);
    while (!feof(f)) {
        getline(&buffer, &l, f);
        size++;
    }
    delete[] buffer;
    rewind(f);
    return size;
}

float last;
void printTable(row *table, int size) {
    printf("-----------------------------------------------------------\n");
    printf("| Key |                       Data                        |\n");
    printf("-----------------------------------------------------------\n");
    for (int i = 0; i < size; ++i) {
        printf("|%f | %s", table[i].key, table[i].string);
        if ((last == table[i].key) && ((printFlag == 1))) printf("\n");
        if ((i == (size - 1)) && (printFlag == 0)) {
            last = table[i].key;
            printFlag = 1;
            printf("\n");
        }
        printf("-----------------------------------------------------------\n");
    }
    printf("\n");
}

row *makeTable(FILE *keys, FILE *value, int size) {
    row *table = new row[size];
    for (int i = 0; i < size; ++i) {
        table[i].string = new char[128];
    }
    for (int i = 0; i < size; ++i) {
        fscanf(keys, "%f", &table[i].key);
        fgets(table[i].string, 128, value);
    }
    return table;
```

```
}

int binSearch(row *tab, float key, int size) {
    int l = 0, r = size - 1, mid;
    while (l <= r) {
        mid = (r + l) / 2;
        if (key == tab[mid].key) return mid;
        else if (key < tab[mid].key) r = mid;
        else if (key > tab[mid].key) l = mid + 1;
    }
    return -1;
}

void swapStrings(row *table, int a, int b) {
    row tmp;
    tmp = table[a];
    table[a] = table[b];
    table[b] = tmp;
}

void mixTable(row *tab, int size) {
    sortFlag = 0;
    int a, b;
    for (int i = 0; i < size; ++i) {
        a = rand() % size;
        b = rand() % size;
        swapStrings(tab, a, b);
    }
}

void reverseStrings(row *table, int size) {
    sortFlag = 0;
    for (int i = 0, j = size - 1; i < j; ++i, --j) swapStrings(table, i, j);
}

void sift(row *tab, int i, int bot) {
    int max;
    int flag = 0;
    while ((i * 2 <= bot) && (!flag)) {
        if (i * 2 == bot) max = i * 2;
        else if (tab[i * 2].key > tab[i * 2 + 1].key) max = i * 2;
        else max = i * 2 + 1;
        if (tab[i].key < tab[max].key) {
            swapStrings(tab, i, max);
            i = max;
        } else flag = 1;
    }
}

void heapSort(row *tab, int size) {
    for (int i = size / 2; i >= 0; --i) {
        sift(tab, i, size - 1);
    }
    for (int i = size - 1; i >= 1; --i) {
        swapStrings(tab, 0, i);
        sift(tab, 0, i - 1);
    }
}

void sortTable(row *tab, int size) {
    sortFlag = 1;
    heapSort(tab, size);
}
```

```
int main() {
    srand(time(0));
    FILE *firstFile = fopen("IN1", "r");
    if (firstFile == NULL) {
        printf("Can not open file 1\n");
        return 0;
    }
    FILE *secondFile = fopen("IN2", "r");
    if (secondFile == NULL) {
        printf("Can not open file 2\n");
        return 0;
    }
    int n = getSize(firstFile), action;
    row *tab = makeTable(firstFile, secondFile, n);
    fclose(firstFile);
    fclose(secondFile);
    printTable(tab, n);
    while (true) {
        printf("Menu\n"
               "1. Print table\n"
               "2. Mix table\n"
               "3. Reverse table\n"
               "4. Binary search\n"
               "5. Sort table\n"
               "0. Exit\n"
               "Choose an action =>");
        scanf("%d", &action);

        switch (action) {
            case 1: {
                printTable(tab, n);
                break;
            }
            case 2: {
                mixTable(tab, n);
                break;
            }
            case 3: {
                reverseStrings(tab, n);
                break;

            }
            case 4: {
                if (sortFlag) {
                    printf("Enter the key: ");
                    float key;
                    scanf("%f", &key);
                    int search;
                    printf("\n");
                    search = binSearch(tab, key, n);
                    if (search == -1) {
                        printf("Element with such key is not found!\n");
                        break;
                    }
                    printf("Found the string:\n %s", tab[search].string);
                } else
                    printf("Table is not sorted!\n");
                break;
            }
            case 5: {
                sortTable(tab, n);
                break;
```

```
            }
        }
    }
}


isitmuse@isitmuse:~/lab/secondSem/kp9$ cat IN1
0.73
1.00
2.32
3.54
3.56
5.44
6.63
7.22
8.63
9.68
10.88
10.89
12.22
13.41
14.65
15.33
isitmuse@isitmuse:~/lab/secondSem/kp9$ cat IN2
Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et
In arcu cursus euismod quis viverra.
Sagittis eu volutpat odio facilisis mauris.
Commodo elit at imperdiet dui accumsan sit amet nulla.
In fermentum et sollicitudin ac orci phasellus egestas.
Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et
Morbi tincidunt augue interdum velit euismod.
At varius vel pharetra vel turpis nunc eget lorem.
Cras fermentum odio eu feugiat.
Lectus sit amet est placerat.
Quis enim lobortis scelerisque fermentum dui faucibus in.
Amet facilisis magna etiam tempor orci eu lobortis elementum.
In est ante in nibh mauris cursus mattis molestie a.
Facilisis volutpat est velit egestas dui id.
Netus et malesuada fames ac turpis egestas integer eget.
Volutpat lacus laoreet non curabitur.
isitmuse@isitmuse:~/lab/secondSem/kp9$ g++ -o kp9 kp9.cpp
isitmuse@isitmuse:~/lab/secondSem/kp9$ ./kp9
-------------------------------------------------------------
| Key |                      Data                       |
-------------------------------------------------------------
|0.730000 | Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt u
-------------------------------------------------------------
|1.000000 | In arcu cursus euismod quis viverra.
-------------------------------------------------------------
|2.320000 | Sagittis eu volutpat odio facilisis mauris.
-------------------------------------------------------------
|3.540000 | Commodo elit at imperdiet dui accumsan sit amet nulla.
-------------------------------------------------------------
|3.560000 | In fermentum et sollicitudin ac orci phasellus egestas.
-------------------------------------------------------------
|5.440000 | Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt u
-------------------------------------------------------------
|6.630000 | Morbi tincidunt augue interdum velit euismod.
-------------------------------------------------------------
|7.220000 | At varius vel pharetra vel turpis nunc eget lorem.
-------------------------------------------------------------
|8.630000 | Cras fermentum odio eu feugiat.
-------------------------------------------------------------
```

```
|9.680000 | Lectus sit amet est placerat.
------------------------------------------------------------
|10.880000 | Quis enim lobortis scelerisque fermentum dui faucibus in.
------------------------------------------------------------
|10.890000 | Amet facilisis magna etiam tempor orci eu lobortis elementum.
------------------------------------------------------------
|12.220000 | In est ante in nibh mauris cursus mattis molestie a.
------------------------------------------------------------
|13.410000 | Facilisis volutpat est velit egestas dui id.
------------------------------------------------------------
|14.650000 | Netus et malesuada fames ac turpis egestas integer eget.
------------------------------------------------------------
|15.330000 | Volutpat lacus laoreet non curabitur.
------------------------------------------------------------

Menu
1. Print table
2. Mix table
3. Reverse table
4. Binary search
5. Sort table
0. Exit
Choose an action =>2
Menu
1. Print table
2. Mix table
3. Reverse table
4. Binary search
5. Sort table
0. Exit
Choose an action =>1
------------------------------------------------------------
| Key |                        Data                         |
------------------------------------------------------------
|8.630000 | Cras fermentum odio eu feugiat.
------------------------------------------------------------
|1.000000 | In arcu cursus euismod quis viverra.
------------------------------------------------------------
|5.440000 | Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt u
------------------------------------------------------------
|9.680000 | Lectus sit amet est placerat.
------------------------------------------------------------
|12.220000 | In est ante in nibh mauris cursus mattis molestie a.
------------------------------------------------------------
|10.890000 | Amet facilisis magna etiam tempor orci eu lobortis elementum.
------------------------------------------------------------
|7.220000 | At varius vel pharetra vel turpis nunc eget lorem.
------------------------------------------------------------
|2.320000 | Sagittis eu volutpat odio facilisis mauris.
------------------------------------------------------------
|3.560000 | In fermentum et sollicitudin ac orci phasellus egestas.
------------------------------------------------------------
|14.650000 | Netus et malesuada fames ac turpis egestas integer eget.
------------------------------------------------------------
|13.410000 | Facilisis volutpat est velit egestas dui id.
------------------------------------------------------------
|15.330000 | Volutpat lacus laoreet non curabitur.
------------------------------------------------------------
|6.630000 | Morbi tincidunt augue interdum velit euismod.
------------------------------------------------------------
|10.880000 | Quis enim lobortis scelerisque fermentum dui faucibus in.
------------------------------------------------------------
|3.540000 | Commodo elit at imperdiet dui accumsan sit amet nulla.
```

```
------------------------------------------------------------
|0.730000 | Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt u
------------------------------------------------------------

Menu
1. Print table
2. Mix table
3. Reverse table
4. Binary search
5. Sort table
0. Exit
Choose an action =>3
Menu
1. Print table
2. Mix table
3. Reverse table
4. Binary search
5. Sort table
0. Exit
Choose an action =>1
------------------------------------------------------------
| Key |                       Data                         |
------------------------------------------------------------
|0.730000 | Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt u
------------------------------------------------------------
|3.540000 | Commodo elit at imperdiet dui accumsan sit amet nulla.
------------------------------------------------------------
|10.880000 | Quis enim lobortis scelerisque fermentum dui faucibus in.
------------------------------------------------------------
|6.630000 | Morbi tincidunt augue interdum velit euismod.
------------------------------------------------------------
|15.330000 | Volutpat lacus laoreet non curabitur.
------------------------------------------------------------
|13.410000 | Facilisis volutpat est velit egestas dui id.
------------------------------------------------------------
|14.650000 | Netus et malesuada fames ac turpis egestas integer eget.
------------------------------------------------------------
|3.560000 | In fermentum et sollicitudin ac orci phasellus egestas.
------------------------------------------------------------
|2.320000 | Sagittis eu volutpat odio facilisis mauris.
------------------------------------------------------------
|7.220000 | At varius vel pharetra vel turpis nunc eget lorem.
------------------------------------------------------------
|10.890000 | Amet facilisis magna etiam tempor orci eu lobortis elementum.
------------------------------------------------------------
|12.220000 | In est ante in nibh mauris cursus mattis molestie a.
------------------------------------------------------------
|9.680000 | Lectus sit amet est placerat.
------------------------------------------------------------
|5.440000 | Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt u
------------------------------------------------------------
|1.000000 | In arcu cursus euismod quis viverra.
------------------------------------------------------------
|8.630000 | Cras fermentum odio eu feugiat.
------------------------------------------------------------

Menu
1. Print table
2. Mix table
3. Reverse table
4. Binary search
5. Sort table
0. Exit
```

```
Choose an action =>4
Table is not sorted!
Menu
1. Print table
2. Mix table
3. Reverse table
4. Binary search
5. Sort table
0. Exit
Choose an action =>5
Menu
1. Print table
2. Mix table
3. Reverse table
4. Binary search
5. Sort table
0. Exit
Choose an action =>4
Enter the key: 1

Found the string:
 In arcu cursus euismod quis viverra.
Menu
1. Print table
2. Mix table
3. Reverse table
4. Binary search
5. Sort table
0. Exit
Choose an action =>1
------------------------------------------------------------
| Key |                        Data                         |
------------------------------------------------------------
|0.730000 | Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt u
------------------------------------------------------------
|1.000000 | In arcu cursus euismod quis viverra.
------------------------------------------------------------
|2.320000 | Sagittis eu volutpat odio facilisis mauris.
------------------------------------------------------------
|3.540000 | Commodo elit at imperdiet dui accumsan sit amet nulla.
------------------------------------------------------------
|3.560000 | In fermentum et sollicitudin ac orci phasellus egestas.
------------------------------------------------------------
|5.440000 | Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt u
------------------------------------------------------------
|6.630000 | Morbi tincidunt augue interdum velit euismod.
------------------------------------------------------------
|7.220000 | At varius vel pharetra vel turpis nunc eget lorem.
------------------------------------------------------------
|8.630000 | Cras fermentum odio eu feugiat.
------------------------------------------------------------
|9.680000 | Lectus sit amet est placerat.
------------------------------------------------------------
|10.880000 | Quis enim lobortis scelerisque fermentum dui faucibus in.
------------------------------------------------------------
|10.890000 | Amet facilisis magna etiam tempor orci eu lobortis elementum.
------------------------------------------------------------
|12.220000 | In est ante in nibh mauris cursus mattis molestie a.
------------------------------------------------------------
|13.410000 | Facilisis volutpat est velit egestas dui id.
------------------------------------------------------------
|14.650000 | Netus et malesuada fames ac turpis egestas integer eget.
------------------------------------------------------------
```

```
|15.330000 | Volutpat lacus laoreet non curabitur.
------------------------------------------------------------

Menu
1. Print table
2. Mix table
3. Reverse table
4. Binary search
5. Sort table
0. Exit
Choose an action =>0
isitmuse@isitmuse:~/lab/secondSem/kp9$
```

9. Дневник отладки должен содержать дату и время сеансов отладки, и основные ошибки (ошибки в сценарии и программе, не стандартные операции) и краткие комментарии к ним. В дневнике отладки приводятся сведения об использовании других ЭВМ, существенном участии преподавателя и других лиц в написании и отладке программы.

№	Лаб или дом.	Дата	Время	Событие	Действие по исправлению	Примечание

10. Замечание автора по существу работы _____
_____

11. Выводы Я научился использовать сортировки и бинарный поиск.

Недочеты, допущенные при выполнении задания, могут быть устранены следующим образом _____
_____

Подпись студента _____