Отчет по лабораторной работе № IX по курсу
Алгоритмы и структуры данных

Студент группы М8О-103Б-22 Ахметшин Булат Рамилевич, № по списку 2

Контакты www, e-mail, icq, skype ahmbulat04@yandex.ru

Работа выполнена: 05.05.2023 г.

Преподаватель: доцент каф. 806 Никулин С.П.

Входной контроль знаний с оценкой _____

Отчет сдан « » _____ 202 __ г., итоговая оценка ___

Подпись преподавателя _____

1. **Тема:** Сортировка и поиск. _____

2. **Цель работы:** Составить программу на языке Си с использованием процедур и функций для сортировки таблицы заданным методом и двоичного поиска по ключу в таблице. _____

3. **Задание** (*вариант № 2 - {2, 8}*): Сортировка - линейный выбор с подсчетом, таблица - ключ из строки и целого, 32 байта на ключ, хранение данных и ключей вместе, минимальное число элементов - 18. _____

4. **Оборудование** (*лабораторное*):
ЭВМ _____, процессор _____, имя узла сети _____ с ОП _____ Мб,
НМД _____ Мб. Терминал _____ адрес _____. Принтер _____
Другие устройства _____

Оборудование ПЭВМ студента, если использовалось:
Процессор Intel(R) Core(TM) i7-10510U____ с ОП 8 ГБ____ НМД SSD 512 ГБ____ . Монитор Встроенный 1920x1080
Другие устройства _____

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

Программное обеспечение ЭВМ студента, если использовалось:
Операционная система семейства UNIX_____, наименование Ubuntu_____ версия 22.04_____
интерпретатор команд GNU bash_____ версия 5.1.16_____
Система программирования Visual Studio Code_____ версия 1.77.3_____
Редактор текстов Sublime Text 3_____ версия 3211_____
Утилиты операционной системы Стандартные утилиты OS Linux_____
Прикладные системы и программы Редактор текста nano._____

Местонахождение и имена файлов программ и данных на домашнем компьютере
/home/bulat/Studying/prprm/cr/IX/

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

Для реализации сортировки линейной выборкой будут учитываться только целача часть значения ключей, т.к. сам алгоритм работает с таким типом ключей.
В качестве тестовых примеров я возьму список сотрудников некой организации, ключом сотрудника будет департамент и индекс, значением - имя сотрудника.

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

Составить makefile, написать тестовые файлы, релизовать таблицу и функции обработки и сортировки, отладить, составить протокол.

Пункты 1-7 отчета составляются строго до начала лабораторной работы.
Допущен к выполнению работы. **Подпись преподавателя** _____

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

```
bulat@bulat-Swift-SF314-58:~/Studying/prprm/cr/IV$ script logs/proto1
Script started, output log file is 'logs/proto1'.
bulat@bulat-Swift-SF314-58:~/Studying/prprm/cr/IV$ ls
logs    makefile             sorted.txt  tex
main.c  reversed_sorted.txt  table.h     unsorted.txt
bulat@bulat-Swift-SF314-58:~/Studying/prprm/cr/IV$ cat makefile
CC = gcc
CFLAGS = -std=c99 -Wall -Wextra -Wformat=0

main:
$(CC) $(CFLAGS) -o main main.c
debug:
$(CC) $(CFLAGS) -g -o main main.c

clean:
rm -f *.o mainbulat@bulat-Swift-SF314-58:~/Studying/prprm/cr/IV$ cat table.h
#ifndef TABLE_H
#define TABLE_H

#include <inttypes.h>
#include <stdbool.h>
#include <stdio.h>
#include <stdint.h>
#include <string.h>
#include <stdlib.h>

#define STRING_KEY_CAP 28

typedef uint64_t size_t;

typedef struct {
    char key_s[STRING_KEY_CAP];
    int key_int;
} complex_key;

typedef struct {
    complex_key key;
    char *value;
} item;

typedef struct {
    item *rows;
    uint64_t count;
    uint64_t max_key;
} table;


table table_alloc();

table table_copy(table t);
void table_push(table *t, char *key_s, int key_int, char *value);
table table_quick_sort(table t);

char *table_binary_search(table t, char *key_s, int key_int);

void table_print(table t);

void table_dealloc(table *t);


#endif // TABLE_H



#ifdef TABLE_IMPLEMENTATION

table table_alloc() {
    table t;
    t.rows = (item*) calloc(0, sizeof(item));
    t.count = 0;
    t.max_key = 0;
    return t;
}


item item_copy(item* const a) {
    complex_key ck = {
        .key_int = a->key.key_int
    };
```

```
    for (int i = 0; i < STRING_KEY_CAP && i < (int) strlen(a->key.key_s); ++i) {
        ck.key_s[i] = a->key.key_s[i];
    }

    char *value = (char*) calloc(256, sizeof(char));
    strcpy(value, a->value);

    item i = {
        .key = ck,
        .value = value
    };
    return i;
}

table table_copy(table t) {
    table temp = table_alloc();

    for (uint64_t i = 0; i < t.count; ++i) {
        char *value = (char*) calloc(256, sizeof(char));
        strcpy(value, t.rows[i].value);
        table_push(&temp, t.rows[i].key.key_s, t.rows[i].key.key_int, value);
    }

    return temp;
}

void table_push(table *t, char *key_s, int key_int, char *value) {
    complex_key ck = {
        .key_int = key_int
    };

    for (int i = 0; i < STRING_KEY_CAP && i < (int) strlen(key_s); ++i) {
        ck.key_s[i] = key_s[i];
    }

    item i = {
        .key = ck,
        .value = value
    };

    if (t->max_key < key_int) {
        t->max_key = key_int;
    }

    t->rows = (item*) realloc(t->rows, sizeof(item) * (t->count + 1));
    t->count++;

    t->rows[t->count - 1] = i;
}

table counting_sort(table* const A) {
    size_t n = A->count, k = A->max_key;

    size_t* P = (size_t*)calloc(k, sizeof(size_t));

    for (size_t i = 0; i < n; ++i) {
        ++P[A->rows[i].key.key_int - 1];
    }
    for (size_t i = 1; i < k; ++i) {
        P[i] += P[i - 1];
    }
    for (size_t i = k; i > 1; --i) {
        P[i - 1] = P[i - 2];
    }
    P[0] = 0;

    table B;
    B.rows = (item*)calloc(n, sizeof(item));

    for (size_t i = 0; i < n; ++i) {
        size_t p = A->rows[i].key.key_int;
        item t = item_copy(&(A->rows[i]));
        B.rows[P[p - 1]] = t;
        ++P[p - 1];
    }

    B.count = A->count;
    B.max_key = A->max_key;

    return B;
}

void _table_quick_sort(table *t, uint64_t l, uint64_t r) {
    char *pivot_value = t->rows[l].value;
```

```
    complex_key pivot_key = t->rows[l].key;
    uint64_t l_init = l, r_init = r;

    while (l < r) {
        while (
            (
                strcmp(pivot_key.key_s, t->rows[r].key.key_s) < 0 ||
                (
                    strcmp(pivot_key.key_s, t->rows[r].key.key_s) == 0 &&
                    pivot_key.key_int <= t->rows[r].key.key_int
                )
            ) && (l < r)
        ) r--;

        if (l != r) {
            t->rows[l].value = t->rows[r].value;
            t->rows[l].key = t->rows[r].key;
            l++;
        }

        while (
            strcmp(pivot_key.key_s, t->rows[l].key.key_s) > 0 &&
            (l < r)
        ) l++;

        if (l != r) {
            t->rows[r].value = t->rows[l].value;
            t->rows[r].key = t->rows[l].key;
            r--;
        }
    }

    t->rows[l].key = pivot_key;
    t->rows[l].value = pivot_value;

    uint64_t pivot = l;
    l = l_init;
    r = r_init;

    if (l < pivot) _table_quick_sort(t, l, pivot - 1);
    if (r > pivot) _table_quick_sort(t, pivot + 1, r);
}

// sort keys by elements
table table_quick_sort(table t) {
    table temp = table_copy(t);

    _table_quick_sort(&temp, 0, temp.count - 1);

    return temp;
}


char *table_binary_search(table t, char *key_s, int key_int) {
    int64_t l = 0, r = t.count - 1, m = (l + r)/2;

    while (l <= r) {
        m = (l + r)/2;

        char *curr_key_s = t.rows[m].key.key_s;
        int curr_key_int = t.rows[m].key.key_int;

        if (strcmp(key_s, curr_key_s) == 0 && key_int == curr_key_int) {
            return t.rows[m].value;
        }

        if ((key_int >= curr_key_int)) {
            l = m + 1;
        } else {
            r = m - 1;
        }
    }

    return NULL;
}


void table_print(table t) {
    printf(
        "| Key%*s| Value\n", STRING_KEY_CAP + 11 - 3, ""
    );

    for (uint64_t i = 0; i < t.count; ++i) {
        printf("| %*s%11d| %s\n", STRING_KEY_CAP, t.rows[i].key.key_s, t.rows[i].key.key_int, t.rows[i].value);
    }
```

```
}


void table_dealloc(table *t) {
    for (uint64_t i = 0; i < t->count; ++i) {
        t->rows[i].key.key_s[0] = '\0';
        t->rows[i].key.key_int = 0;
        free(t->rows[i].value);
    }

    free(t->rows);
}

#endif // TABLE_IMPLEMENTATIONbulat@bulat-Swift-SF314-58:~/Studying/prprm/cr/IV$ cat main.c
#define TABLE_IMPLEMENTATION
#include "table.h"


int main() {
    table t = table_alloc();

    uint64_t count = 0;
    scanf("%ld", &count);

    printf("Table: ");
    for (uint64_t i = 0; i < count; ++i) {

        char *key_s = (char*) calloc(256, sizeof(char));
        scanf("%s", key_s);

        int key_int = 0;
        scanf("%d", &key_int);

        char *value = (char*) calloc(256, sizeof(char));
        scanf("%s", value);

        table_push(&t, key_s, key_int, value);

        free(key_s);
    }
    printf("\n");

    table_print(t);
    printf("\n");
    table s = counting_sort(&t);
    table_print(s);

    int search = 1;
    while (search) {
        char *key_s = (char*) calloc(256, sizeof(char));
        int key_int = 0;
        scanf("%s", key_s);
        scanf("%d", &key_int);
        printf("Search for:(%s, %d)\n", key_s, key_int);

        printf("Found string: %s\n", table_binary_search(s, key_s, key_int));
        free(key_s);

        printf("Do you want to continue the search? (0/1): ");
        scanf("%d", &search);
    }

    printf("\n");

    table_dealloc(&t);
}bulat@bulat-Swift-SF314-58:~/Studying/prprm/cr/IV$ make
gcc -std=c99 -Wall -Wextra -Wformat=0 -o main main.c
In file included from main.c:2:
table.h: In function ‘table_push’:
table.h:105:20: warning: comparison of integer expressions of different signedness: ‘uint64_t’ {aka ‘long unsigned int’} and
  105 |     if (t->max_key < key_int) {
      |                    ^
bulat@bulat-Swift-SF314-58:~/Studying/prprm/cr/IV$ ./main <sorted.txt
Table:
| Key                                 | Value
|     TransportationDepartment          1| Mike
|             LegalDepartment           2| George
|     TransportationDepartment          3| Jacob
|     TransportationDepartment          4| Kevin
|           DevelopmentCentre           5| Alice
|             LegalDepartment           6| Ioan
|           DevelopmentCentre           7| Frida
|           DevelopmentCentre           8| Alice
|         FinancialDepartment           9| Alex
|           DevelopmentCentre          10| Morty
```

```
|          FinancialDepartment         11| Rick
|          FinancialDepartment         12| Daniel
|          FinancialDepartment         13| Belle
|          FinancialDepartment         14| Michel
|            DevelopmentCentre         15| Patrix
|     TransportationDepartment         16| Paul
|            DevelopmentCentre         17| Jane
|              LegalDepartment         18| Freya

| Key                                   | Value
|     TransportationDepartment          1| Mike
|              LegalDepartment          2| George
|     TransportationDepartment          3| Jacob
|     TransportationDepartment          4| Kevin
|            DevelopmentCentre          5| Alice
|              LegalDepartment          6| Ioan
|            DevelopmentCentre          7| Frida
|            DevelopmentCentre          8| Alice
|          FinancialDepartment          9| Alex
|            DevelopmentCentre         10| Morty
|          FinancialDepartment         11| Rick
|          FinancialDepartment         12| Daniel
|          FinancialDepartment         13| Belle
|          FinancialDepartment         14| Michel
|            DevelopmentCentre         15| Patrix
|     TransportationDepartment         16| Paul
|            DevelopmentCentre         17| Jane
|              LegalDepartment         18| Freya
Search for:(FinancialDepartment, 11)
Found string: Rick
Do you want to continue the search? (0/1): Search for:(LegalDepartment, 6)
Found string: Ioan
Do you want to continue the search? (0/1): Search for:(TransportationDepartment, 3)
Found string: Jacob
Do you want to continue the search? (0/1):
bulat@bulat-Swift-SF314-58:~/Studying/prprm/cr/IV$ ./main <unsorted.txt
Table:
| Key                                   | Value
|     TransportationDepartment          1| Mike
|     TransportationDepartment          4| Kevin
|            DevelopmentCentre          8| Alice
|              LegalDepartment         18| Freya
|          FinancialDepartment          9| Alex
|            DevelopmentCentre         15| Patrix
|            DevelopmentCentre          5| Alice
|            DevelopmentCentre          7| Frida
|              LegalDepartment          2| George
|     TransportationDepartment         16| Paul
|            DevelopmentCentre         10| Morty
|     TransportationDepartment          3| Jacob
|          FinancialDepartment         13| Belle
|          FinancialDepartment         12| Daniel
|              LegalDepartment          6| Ioan
|          FinancialDepartment         14| Michel
|          FinancialDepartment         11| Rick
|            DevelopmentCentre         17| Jane

| Key                                   | Value
|     TransportationDepartment          1| Mike
|              LegalDepartment          2| George
|     TransportationDepartment          3| Jacob
|     TransportationDepartment          4| Kevin
|            DevelopmentCentre          5| Alice
|              LegalDepartment          6| Ioan
|            DevelopmentCentre          7| Frida
|            DevelopmentCentre          8| Alice
|          FinancialDepartment          9| Alex
|            DevelopmentCentre         10| Morty
|          FinancialDepartment         11| Rick
|          FinancialDepartment         12| Daniel
|          FinancialDepartment         13| Belle
|          FinancialDepartment         14| Michel
|            DevelopmentCentre         15| Patrix
|     TransportationDepartment         16| Paul
|            DevelopmentCentre         17| Jane
|              LegalDepartment         18| Freya
Search for:(FinancialDepartment, 11)
Found string: Rick
Do you want to continue the search? (0/1): Search for:(LegalDepartment, 6)
Found string: Ioan
Do you want to continue the search? (0/1): Search for:(TransportationDepartment, 3)
Found string: Jacob
Do you want to continue the search? (0/1):
bulat@bulat-Swift-SF314-58:~/Studying/prprm/cr/IV$ ./main <reversed_sorted.txt
Table:
```

```
| Key                                  | Value
|              LegalDepartment        18| Freya
|            DevelopmentCentre        17| Jane
|     TransportationDepartment        16| Paul
|            DevelopmentCentre        15| Patrix
|          FinancialDepartment        14| Michel
|          FinancialDepartment        13| Belle
|          FinancialDepartment        12| Daniel
|          FinancialDepartment        11| Rick
|            DevelopmentCentre        10| Morty
|          FinancialDepartment         9| Alex
|            DevelopmentCentre         8| Alice
|            DevelopmentCentre         7| Frida
|              LegalDepartment         6| Ioan
|            DevelopmentCentre         5| Alice
|     TransportationDepartment         4| Kevin
|     TransportationDepartment         3| Jacob
|              LegalDepartment         2| George
|     TransportationDepartment         1| Mike

| Key                                  | Value
|     TransportationDepartment         1| Mike
|              LegalDepartment         2| George
|     TransportationDepartment         3| Jacob
|     TransportationDepartment         4| Kevin
|            DevelopmentCentre         5| Alice
|              LegalDepartment         6| Ioan
|            DevelopmentCentre         7| Frida
|            DevelopmentCentre         8| Alice
|          FinancialDepartment         9| Alex
|            DevelopmentCentre        10| Morty
|          FinancialDepartment        11| Rick
|          FinancialDepartment        12| Daniel
|          FinancialDepartment        13| Belle
|          FinancialDepartment        14| Michel
|            DevelopmentCentre        15| Patrix
|     TransportationDepartment        16| Paul
|            DevelopmentCentre        17| Jane
|              LegalDepartment        18| Freya
Search for:(FinancialDepartment, 11)
Found string: Rick
Do you want to continue the search? (0/1): Search for:(LegalDepartment, 6)
Found string: Ioan
Do you want to continue the search? (0/1): Search for:(TransportationDepartment, 3)
Found string: Jacob
Do you want to continue the search? (0/1):
```

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

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

10. **Замечания автора** по существу работы: _____

11. **Выводы:** в ходе этой лабораторной работы я получил опыт реализации таблиц с комплесными ключами и функций для их обработки и сортировки.

12. Недочёты при выполнении задания могут быть устранены следующим образом: _____

Подпись студента _____