ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ

**Федеральное государственное автономное образовательное учреждение
высшего образования
«Национальный исследовательский университет
«Высшая школа экономики»**

Московский институт электроники и математики им. А.Н. Тихонова

Департамент прикладной математики

Отчёт
по лабораторной работе №9
по курсу «Алгоритмизация и программирование»

ФИО студента	Номер группы	Дата
Вязов Глеб Дмитриевич	БПМ-231	02.03.2024

Москва, 2023

Задание (вариант №7)

1. Реализовать указанные в варианте алгоритмы сортировки для массива объектов в соответствии с вариантом. Структура объекта описана в варианте ЛР8. Определить функции сравнения объектов по следующему принципу: приоритет сравнения определяется порядком поля в структуре, т.е. если структурный тип X содержит (в описании варианта) поля с именами a,b,c,d, то при сравнении двух объектов типа X надо сравнить поля с именем a, в случае их равенства сравнить поля b и т.д.

2. Каждый алгоритм сортировки должен поддерживать сортировку в обоих направлениях: по неубыванию и по невозрастанию.

3. Выбор и запуск требуемого алгоритма и направления сортировки осуществляется через меню на этапе выполнения.

4. Провести сортировку каждым алгоритмом массивов следующих размеров: 100, 1000, 5000, 10000, 20000, 50000, 100000. Засечь (программно) время сортировки каждым алгоритмом. По полученным точкам построить графики зависимости времени сортировки от размерности массива для каждого из алгоритмов сортировки на одной оси координат. Полученные графики включить в отчет к работе.

Сортировки, которые нужно реализовать:

1. Сортировка вставками

2. Сортировка слиянием

3. Быстрая сортировка

Решение

Листинг 1: structs.h

```h
#ifndef HW9_STRUCTS_H
#define HW9_STRUCTS_H

# define N 50

struct Sportsmen {
    char fio[N];  // ФИО
    int age;      // Возраст
    int height;   // Рост
    int weight;   // Вес
    char type[N]; // Вид спорта
    char rank[N]; // Спотртивное звание
};

void printSportsmen(struct Sportsmen sportsmen);
void printArrayOfSportsmen(struct Sportsmen sportsmen[],
   int len);
int compareSportsmens(struct Sportsmen sportsmen1, struct
   Sportsmen sportsmen2);
struct Sportsmen* generateSportmens(int count);
int random(int min, int max);

#endif
```

Листинг 2: sorting.c

```
#include <stdlib.h>
#include "repository.c"

/*
 * Длявсехсортировок : параметр ascending
    отвечаетзапорядоксортировки
 * Если ascending = 1, тосортировкаповозрастанию
 * Если ascending = -1, тосортировкапоубыванию
 */

// Сортировка вставками. Время: O(n**2). Память: O(1)
void insertSort(struct Sportsmen array[], int len, int
    ascending) {
    struct Sportsmen value;
    int j;

    for (int i=1; i<len; i++) {
        value = array[i];

        for (j=i-1; j >= 0; j--) {
            if (ascending*compareSportsmens(array[j], value
                ) == 1) {
                array[j+1] = array[j];
            } else {
                break;
            }
        }

        array[j+1] = value;
    }
}

// Вспомогательная функция для сортировки слиянием
void merge(struct Sportsmen array[], int l, int m, int r,
    int ascending) {
    int len1 = m - l + 1;
    int len2 = r - m;
    struct Sportsmen* left = calloc(len1, sizeof(struct
        Sportsmen)); // [l,
```

```
    m]
35      struct Sportsmen* right = calloc(len2, sizeof(struct
            Sportsmen)); // [m+1,
        r]
36      int i = l, i1=0, i2=0; // i – индекс для array, i1 – индекс для left,
            i2 – индекс для right
37
38      // Заполняем левые и правые части массивов элементами исходного
39      for (int k=l; k<=m; k++) {
40          left[k-l] = array[k];
41      }
42      for (int k=m+1; k<=r; k++) {
43          right[k-(m+1)] = array[k];
44      }
45
46      // Алгоритм сортировки
47      while (i1 < len1 && i2 < len2) {
48          if (ascending*compareSportsmens(right[i2], left[i1
                ]) == 1) { // right[i2] >=
            left[i1]
49              array[i] = left[i1];
50              i1++;
51          } else {
52              array[i] = right[i2];
53              i2++;
54          }
55          i++;
56      }
57
58      // Один из индексов может не дойти до конца
59      while (i1 < len1) {
60          array[i] = left[i1];
61          i++;
62          i1++;
63      }
64
65      while (i2 < len2) {
66          array[i] = right[i2];
67          i++;
68          i2++;
69      }
```

```
70
71      free(left);
72      free(right);
73  }
74
75  // Сортировка слиянием. Время: O(nlog n). Память: O(n)
76  // Сортируемый диапозон: array[l], ..., array[r] (все границы включены)
77  void mergeSort(struct Sportsmen array[], int l, int r, int
        ascending) {
78      if (r <= l) {
79          return;
80      }
81
82      int m = l + (r - l) / 2;
83      mergeSort(array, l, m, ascending);
84      mergeSort(array, m+1, r, ascending);
85      merge(array, l, m, r, ascending);
86  }
87
88  // Вспомогательная функция для быстрой сортировки
89  // Меняет местами значения a и b
90  void swap(struct Sportsmen* a, struct Sportsmen* b) {
91      struct Sportsmen temp = *a;
92      *a = *b;
93      *b = temp;
94  }
95
96  // Вспомогательная функция для быстрой сортировки
97  // Возвращает индекс "разделяющего"элемента
98  // Слева от "разделяющего"элемента находятся элементы, которые меньше его,
        а справа, которые больше
99  int partition(struct Sportsmen array[], int l, int r, int
        ascending) {
100     struct Sportsmen pivot = array[l];
101     int i = l, j = r;
102
103     while (i < j) {
104         if (ascending == 1) {
105             // Находим индекс элемента, который больше pivot: pivot >=
                    array[i]
```

```
106             while (compareSportsmens(pivot, array[i]) == 1
                    && i <= r - 1) { i++; }
107             // Находим индекс элемента, который меньше pivot: array[j] >=
                    pivot
108             while (compareSportsmens(array[j], pivot) == 1
                    && j >= l + 1) { j--; }
109         } else {
110             // Находим индекс элемента, который меньше pivot: array[i] >=
                    pivot
111             while (compareSportsmens(array[i], pivot) == 1
                    && i <= r - 1) { i++; }
112             // Находим индекс элемента, который больше pivot: pivot >=
                    array[j]
113             while (compareSportsmens(pivot, array[j]) == 1
                    && j >= l + 1) { j--; }
114         }
115
116         if (i < j) {
117             swap(&array[i], &array[j]);
118         }
119     }
120     // Возвращаем на место разделяющий элемент
121     swap(&array[l], &array[j]);
122     return j;
123 }
124
125 // Быстрая сортировка. Время: O(nlog n). Память: O(1)
126 void quickSort(struct Sportsmen array[], int l, int r, int
        ascending) {
127     if (l >= r) {
128         return;
129     }
130
131     // Выбираем разделяющий элемент
132     int partitionIndex = partition(array, l, r, ascending);
133
134     quickSort(array, l, partitionIndex - 1, ascending);
135     quickSort(array, partitionIndex + 1, r, ascending);
136 }
```

Листинг 3: repository.c

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "structs.h"

#define LEN_FIOS 18
#define LEN_TYPES 15
#define LEN_RANKS 14

const char FIOS[LEN_FIOS][N] = {
        "Иван Иванов", "Павел Артемьев", "Емельяненко Федор",
        "Арнольд Шварцнегер", "Луговой Александр", "Сарычев
            Кирилл", "Джулиус Мэддокс",
        "Тайсон Майк", "Попович Алеша", "Муромец Илья", "
            Добрыня Никитич", "Ронни Колман",
        "Цзю Константин", "Хабиб", "Конор Макгрегор", "Фишер
            Роберт", "Карякин Сергей",
        "Магнус Карлсен"
};
const char TYPES[LEN_TYPES][N] = {
        "Тяжелая атлетика", "ММА", "Бокс", "Самбо", "Дзюдо", "
            Пауэрлифтинг",
        "Легкая атлетика", "АРБ", "Кикбоксинг", "Тайский бокс", "
            БЖЖ",
        "Бодибилдинг", "Шахматы", "Водное поло", "Плавание"
};
const char RANKS[LEN_RANKS][N] = {
        "ЧМ", "МСМК", "ЗМС", "МС", "КМС", "1
            взрослыйразряд ", "2 взрослыйразряд ",
        "3 взрослыйразряд ", "1 юношескийразряд ", "2
            юношескийразряд ", "3 юношескийразряд ",
        "Любитель", "Просто заЗОЖ ", "Олимпиец"
};

// Вывод спортсмена в консоль
void printSportsmen(struct Sportsmen sportsmen) {
    printf("%s, %d, %d, %d, %s, %s\n", sportsmen.fio,
        sportsmen.age, sportsmen.height, sportsmen.weight,
            sportsmen.type, sportsmen.rank);
```

```
32 }
33
34 // Вывод массива спортсменов в консоль
35 void printArrayOfSportsmen(struct Sportsmen sportsmen[],
       int len) {
36     for (int i=0; i<len; i++) {
37         printSportsmen(sportsmen[i]);
38     }
39     printf("\n");
40 }
41
42 // Функция сравнивает двух спортсменов
43 // Возвращает 1, если sportsmen1 >= sportsmen2
44 // Возвращает -1, если sportsmen1 < sportsmen2
45 int compareSportsmens(struct Sportsmen sportsmen1, struct
       Sportsmen sportsmen2) {
46     // Сравнение ФИО
47     int compareFio = strcmp(sportsmen1.fio, sportsmen2.fio)
           ;
48     if (compareFio > 0) {
49         return 1;
50     } else if (compareFio < 0) {
51         return -1;
52     }
53
54     // Сравнение возраста
55     if (sportsmen1.age > sportsmen2.age) {
56         return 1;
57     } else if (sportsmen1.age < sportsmen2.age) {
58         return -1;
59     }
60
61     // Сравнение роста
62     if (sportsmen1.height > sportsmen2.height) {
63         return 1;
64     } else if (sportsmen1.height < sportsmen2.height) {
65         return -1;
66     }
67
68     // Сравнение веса
```

```
69      if (sportsmen1.weight > sportsmen2.weight) {
70          return 1;
71      } else if (sportsmen1.weight < sportsmen2.weight) {
72          return -1;
73      }
74
75      // Сравнение вида спорта
76      int compareType = strcmp(sportsmen1.type, sportsmen2.
            type);
77      if (compareType > 0) {
78          return 1;
79      } else if (compareType < 0) {
80          return -1;
81      }
82
83      // Сравнение спортивного звания
84      int compareRank = strcmp(sportsmen1.rank, sportsmen2.
            rank);
85      if (compareRank > 0) {
86          return 1;
87      } else if (compareRank < 0) {
88          return -1;
89      }
90
91      // Объекты равны
92      return 1;
93  }
94
95  // Генерация пвесво-рандомного числа из диапозона [min; max]
96  int random(int min, int max) {
97      return min + rand()%(max-min);
98  }
99
100 // Функция генерирует массив из спортсменов длиной count
101 struct Sportsmen* generateSportmens(int count) {
102     struct Sportsmen* sportsmens = calloc(count, sizeof(
            struct Sportsmen));
103
104     for (int i=0; i<count; i++) {
105         struct Sportsmen sp;
```

```
106             strcpy(sp.fio, FIOS[random(0, LEN_FIOS-1)]);
107             sp.age = random(18, 60),
108             sp.height = random(160, 210),
109             sp.weight = random(50, 170),
110                     strcpy(sp.type, TYPES[random(0, LEN_TYPES
                        -1)]);
111             strcpy(sp.rank, RANKS[random(0, LEN_RANKS-1)]);
112
113             sportsmens[i] = sp;
114         }
115         return sportsmens;
116 }
```

Листинг 4: hw9.c

```c
#include <stdio.h>
#include <windows.h>
#include "sorting.c"

int main() {
    // Меняем кодировку на UTF-8, чтобы можно было писать на русском
    SetConsoleOutputCP(CP_UTF8);
    // Ввод переменных. Дружественный интерфейс
    printf("Выполнил задание: ВязовГлеб . Группа: БПМ231\n");

    int count, type_sorting, asc;

    printf("Введите длинумассива , которыйбудемсортировать : ");
    scanf("%d", &count);
    struct Sportsmen* sportsmens = generateSportmens(count)
        ;
    printArrayOfSportsmen(sportsmens, count);
    printf("Выберите алгоритмсортировки :\n"
           "1. Сортировкавставками \n"
           "2. Сортировкаслиянием \n"
           "3. Быстраясортировка \n");
    scanf("%d", &type_sorting);
    printf("Выберите направлениесортировки :\n"
           "По возврастанию -- 1\n"
           "По убыванию -- -1\n");
    scanf("%d", &asc);
    switch (type_sorting) {
        case 1:
            insertSort(sportsmens, count, asc);
            break;
        case 2:
            mergeSort(sportsmens, 0, count-1, asc);
            break;
        case 3:
            quickSort(sportsmens, 0, count-1, asc);
            break;
    }
    printArrayOfSportsmen(sportsmens, count);
    free(sportsmens);
```

```
39 }
```

Листинг 5: timing.c

```c
#include <time.h>
#include <windows.h>
#include "sorting.c"

const int sizes[] = {100, 1000, 5000, 10000, 20000, 50000, 100000};

int main() {
    // Меняем кодировку на UTF-8, чтобы можно было писать на русском
    SetConsoleOutputCP(CP_UTF8);

    double timeSpent;
    printf("Сортировка вставками:\n");
    for (int i=0; i<7; i++) {
        struct Sportsmen* sportsmens = generateSportmens(sizes[i]);

        clock_t begin = clock();
        insertSort(sportsmens, sizes[i], 1);
        clock_t end = clock();
        timeSpent = (double)(end - begin) / CLOCKS_PER_SEC;

        printf("Длина массива = %5d, времяработы  = %10.4lf секунд\n", sizes[i], timeSpent);
    }

    printf("\Сортировкап слиянием:\n");
    for (int i=0; i<7; i++) {
        struct Sportsmen* sportsmens = generateSportmens(sizes[i]);

        clock_t begin = clock();
        mergeSort(sportsmens, 0, sizes[i]-1, 1);
        clock_t end = clock();
        timeSpent = (double)(end - begin) / CLOCKS_PER_SEC;

        printf("Длина массива = %5d, времяработы  = %10.4lf секунд\n", sizes[i], timeSpent);
    }
```

```
35
36      printf("\Быстраян сортировка:\n");
37      for (int i=0; i<7; i++) {
38          struct Sportsmen* sportsmens = generateSportmens(
                sizes[i]);
39
40          clock_t begin = clock();
41          quickSort(sportsmens, 0, sizes[i]-1, 1);
42          clock_t end = clock();
43          timeSpent = (double)(end - begin) / CLOCKS_PER_SEC;
44
45          printf("Длина массива = %6d, времяработы  = %10.4lf
                секунд\n", sizes[i], timeSpent);
46      }
47
48      return 0;
49  }
```

Тестирование

```
Выполнил задание: Вязов Глеб. Группа: БПМ231
Введите длину массива, который будем сортировать:5
 Тайсон Майк, 47, 194, 150, Самбо, 3 взрослый разряд
Арнольд Шварцнегер, 18, 172, 154, АРБ, ЧМ
Попович Алеша, 45, 171, 61, Водное поло, 1 юношеский разряд
Карякин Сергей, 36, 201, 134, БЖЖ, 3 юношеский разряд
Арнольд Шварцнегер, 52, 181, 166, Легкая атлетика, 1 взрослый разряд

Выберите алгоритм сортировки:
1. Сортировка вставками
2. Сортировка слиянием
3. Быстрая сортировка
2
Выберите направление сортировки:
По возврастанию -- 1
По убыванию -- -1
1
Арнольд Шварцнегер, 18, 172, 154, АРБ, ЧМ
Арнольд Шварцнегер, 52, 181, 166, Легкая атлетика, 1 взрослый разряд
Карякин Сергей, 36, 201, 134, БЖЖ, 3 юношеский разряд
Попович Алеша, 45, 171, 61, Водное поло, 1 юношеский разряд
Тайсон Майк, 47, 194, 150, Самбо, 3 взрослый разряд
```

```
Введите длину массива, который будем сортировать:5
 Тайсон Майк, 47, 194, 150, Самбо, 3 взрослый разряд
Арнольд Шварцнегер, 18, 172, 154, АРБ, ЧМ
Попович Алеша, 45, 171, 61, Водное поло, 1 юношеский разряд
Карякин Сергей, 36, 201, 134, БЖЖ, 3 юношеский разряд
Арнольд Шварцнегер, 52, 181, 166, Легкая атлетика, 1 взрослый разряд

Выберите алгоритм сортировки:
1. Сортировка вставками
2. Сортировка слиянием
3. Быстрая сортировка
3
Выберите направление сортировки:
По возврастанию -- 1
По убыванию -- -1
-1
Тайсон Майк, 47, 194, 150, Самбо, 3 взрослый разряд
Попович Алеша, 45, 171, 61, Водное поло, 1 юношеский разряд
Карякин Сергей, 36, 201, 134, БЖЖ, 3 юношеский разряд
Арнольд Шварцнегер, 52, 181, 166, Легкая атлетика, 1 взрослый разряд
Арнольд Шварцнегер, 18, 172, 154, АРБ, ЧМ
```