

**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.

## Решение

Листинг 1: structs.h

```
1 #ifndef HW9_STRUCTS_H
2 #define HW9_STRUCTS_H
3
4 # define N 50
5
6 struct Sportsmen {
7     char fio[N];    // ФИО
8     int age;        // Возраст
9     int height;     // Рост
10    int weight;     // Вес
11    char type[N];    // Вид спорта
12    char rank[N];    // Спортивное звание
13 };
14
15 void printSportsmen(struct Sportsmen sportsmen);
16 void printArrayOfSportsmen(struct Sportsmen sportsmen[],
17                             int len);
18 int compareSportsmens(struct Sportsmen sportsmen1, struct
19                        Sportsmen sportsmen2);
20 struct Sportsmen* generateSportmens(int count);
21 int random(int min, int max);
22 #endif
```

Листинг 2: sorting.c

```

1 #include <stdlib.h>
2 #include "repository.c"
3
4 /*
5  * Для всех сортировок : параметр ascending
6  * отвечает за порядок сортировки
7  * Если ascending = 1, то сортировка по возрастанию
8  * Если ascending = -1, то сортировка по убыванию
9  */
10 // Сортировка вставками. Время:  $O(n^2)$ . Память:  $O(1)$ 
11 void insertSort(struct Sportsmen array[], int len, int
12     ascending) {
13     struct Sportsmen value;
14     int j;
15
16     for (int i=1; i<len; i++) {
17         value = array[i];
18
19         for (j=i-1; j >= 0; j--) {
20             if (ascending*compareSportsmens(array[j], value
21                 ) == 1) {
22                 array[j+1] = array[j];
23             } else {
24                 break;
25             }
26         }
27         array[j+1] = value;
28     }
29 }
30 // Вспомогательная функция для сортировки слиянием
31 void merge(struct Sportsmen array[], int l, int m, int r,
32     int ascending) {
33     int len1 = m - l + 1;
34     int len2 = r - m;
35     struct Sportsmen* left = calloc(len1, sizeof(struct
36         Sportsmen)); // l,

```

```

35 |         m]
36 | struct Sportsmen* right = calloc(len2, sizeof(struct
    | Sportsmen)); // [m+1,
    | r]
37 |
38 | int i = l, i1=0, i2=0; // i – индекс для array, i1 – индекс для left,
    | i2 – индекс для right
39 |
40 | // Заполняем левые и правые части массивов элементами исходного
41 | for (int k=l; k<=m; k++) {
42 |     left[k-l] = array[k];
43 | }
44 | for (int k=m+1; k<=r; k++) {
45 |     right[k-(m+1)] = array[k];
46 | }
47 |
48 | // Алгоритм сортировки
49 | while (i1 < len1 && i2 < len2) {
50 |     if (ascending*compareSportsmens(right[i2], left[i1
    | left[i1]
51 |     array[i] = left[i1];
52 |     i1++;
53 | } else {
54 |     array[i] = right[i2];
55 |     i2++;
56 | }
57 | i++;
58 | }
59 |
60 | // Один из индексов может не дойти до конца
61 | while (i1 < len1) {
62 |     array[i] = left[i1];
63 |     i++;
64 |     i1++;
65 | }
66 |
67 | while (i2 < len2) {
68 |     array[i] = right[i2];
69 |     i++;
70 |     i2++;
71 | }

```

```

70
71     free(left);
72     free(right);
73 }
74
75 // Сортировка слиянием. Время: O(n log n). Память: O(n)
76 // Сортируемый диапазон: 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
107             && i <= r - 1) { i++; }
108         // Находим индекс элемента, который меньше pivot: array[j] >=
109         // pivot
110         while (compareSportsmens(array[j] , pivot) == 1
111             && j >= l + 1) { j--; }
112     } else {
113         // Находим индекс элемента, который меньше pivot: array[i] >=
114         // pivot
115         while (compareSportsmens(array[i] , pivot) == 1
116             && i <= r - 1) { i++; }
117         // Находим индекс элемента, который больше pivot: pivot >=
118         // array[j]
119         while (compareSportsmens(pivot , array[j]) == 1
120             && j >= l + 1) { j--; }
121     }
122     if (i < j) {
123         swap(&array[i] , &array[j]);
124     }
125     // Возвращаем на место разделяющий элемент
126     swap(&array[l] , &array[j]);
127     return j;
128 }
129
130 // Быстрая сортировка. Время: O(nlog n). Память: O(1)
131 void quickSort(struct Sportsmen array[], int l, int r, int
132     ascending) {
133     if (l >= r) {
134         return;
135     }
136     // Выбираем разделяющий элемент
137     int partitionIndex = partition(array , l , r , ascending);
138     quickSort(array , l , partitionIndex - 1, ascending);
139     quickSort(array , partitionIndex + 1, r , ascending);
140 }

```

Листинг 3: repository.c

```

1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4 #include "structs.h"
5
6 #define LEN_FIOS 18
7 #define LEN_TYPES 15
8 #define LEN_RANKS 14
9
10 const char FIOS[LEN_FIOS][N] = {
11     "Иван Иванов", "Павел Артемьев", "Емельяненко Федор",
12     "Арнольд Шварцнегер", "Луговой Александр", "Сарычев
        Кирилл", "Джулиус Мэддокс",
13     "Тайсон Майк", "Полович Алеша", "Муромец Илья", "
        Добрыня Никитич", "Ронни Колман",
14     "Цзю Константин", "Хабиб", "Конор Макгрегор", "Фишер
        Роберт", "Карякин Сергей",
15     "Магнус Карлсен"
16 };
17 const char TYPES[LEN_TYPES][N] = {
18     "Тяжелая атлетика", "ММА", "Бокс", "Самбо", "Дзюдо", "
        Пауэрлифтинг",
19     "Легкая атлетика", "АРБ", "Кикбоксинг", "Тайский бокс", "
        БЖЖ",
20     "Бодибилдинг", "Шахматы", "Водное поло", "Плавание"
21 };
22 const char RANKS[LEN_RANKS][N] = {
23     "ЧМ", "МСМК", "ЗМС", "МС", "КМС", "1
        взрослыйразряд ", "2 взрослыйразряд ",
24     "3 взрослыйразряд ", "1 юношескийразряд ", "2
        юношескийразряд ", "3 юношескийразряд ",
25     "Любитель", "Просто заЗОЖ ", "Олимпиец"
26 };
27
28 // Вывод спортсмена в консоль
29 void printSportsmen(struct Sportsmen sportsmen) {
30     printf("%s, %d, %d, %d, %s, %s\n", sportsmen.fio,
        sportsmen.age, sportsmen.height, sportsmen.weight,
31     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.
77                             type);
78     if (compareType > 0) {
79         return 1;
80     } else if (compareType < 0) {
81         return -1;
82     }
83
84     // Сравнение спортивного звания
85     int compareRank = strcmp(sportsmen1.rank, sportsmen2.
86                             rank);
87     if (compareRank > 0) {
88         return 1;
89     } else if (compareRank < 0) {
90         return -1;
91     }
92
93     // Объекты равны
94     return 1;
95 }
96
97 // Генерация псевдо-рандомного числа из диапазона [min; max]
98 int random(int min, int max) {
99     return min + rand()%(max-min);
100 }
101
102 // Функция генерирует массив из спортсменов длиной count
103 struct Sportsmen* generateSportmens(int count) {
104     struct Sportsmen* sportsmens = calloc(count, sizeof(
105         struct Sportsmen));
106
107     for (int i=0; i<count; i++) {
108         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
111                 -1)]);
112         strcpy(sp.rank , RANKS[random(0, LEN_RANKS-1)]);
113         sportsmens[i] = sp;
114     }
115     return sportsmens;
116 }
```

Листинг 4: hw9.c

```

1 #include <stdio.h>
2 #include <windows.h>
3 #include "sorting.c"
4
5 int main() {
6     // Меняем кодировку на UTF-8, чтобы можно было писать на русском
7     SetConsoleOutputCP(CP_UTF8);
8     // Ввод переменных. Дружественный интерфейс
9     printf("Выполнил задание: ВязовГлеб . Группа: БПМ231\n");
10
11     int count, type_sorting, asc;
12
13     printf("Введите длину массива , который будем сортировать : ");
14     scanf("%d", &count);
15     struct Sportsmen* sportsmens = generateSportsmens(count)
16     ;
17     printArrayOfSportsmen(sportsmens, count);
18     printf("Выберите алгоритм сортировки :\n"
19           "1. Сортировка вставками \n"
20           "2. Сортировка слиянием \n"
21           "3. Быстрая сортировка \n");
22     scanf("%d", &type_sorting);
23     printf("Выберите направление сортировки :\n"
24           "По возрастанию — 1\n"
25           "По убыванию — -1\n");
26     scanf("%d", &asc);
27     switch (type_sorting) {
28         case 1:
29             insertSort(sportsmens, count, asc);
30             break;
31         case 2:
32             mergeSort(sportsmens, 0, count-1, asc);
33             break;
34         case 3:
35             quickSort(sportsmens, 0, count-1, asc);
36             break;
37     }
38     printArrayOfSportsmen(sportsmens, count);
39     free(sportsmens);

```

39 | }

Листинг 5: timing.c

```

1 #include <time.h>
2 #include <windows.h>
3 #include "sorting.c"
4
5 const int sizes[] = {100, 1000, 5000, 10000, 20000, 50000,
6     100000};
7
8 int main() {
9     // Меняем кодировку на UTF-8, чтобы можно было писать на русском
10    SetConsoleOutputCP(CP_UTF8);
11
12    double timeSpent;
13    printf("Сортировка вставками:\n");
14    for (int i=0; i<7; i++) {
15        struct Sportsmen* sportsmens = generateSportmens(
16            sizes[i]);
17
18        clock_t begin = clock();
19        insertSort(sportsmens, sizes[i], 1);
20        clock_t end = clock();
21        timeSpent = (double)(end - begin) / CLOCKS_PER_SEC;
22
23        printf("Длина массива = %5d, время работы = %10.4lf\n", sizes[i], timeSpent);
24    }
25
26    printf("\Сортировка слиянием:\n");
27    for (int i=0; i<7; i++) {
28        struct Sportsmen* sportsmens = generateSportmens(
29            sizes[i]);
30
31        clock_t begin = clock();
32        mergeSort(sportsmens, 0, sizes[i]-1, 1);
33        clock_t end = clock();
34        timeSpent = (double)(end - begin) / CLOCKS_PER_SEC;
35
36        printf("Длина массива = %5d, время работы = %10.4lf\n", sizes[i], timeSpent);
37    }
38 }

```

```

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