

Национальный исследовательский ядерный университет «МИФИ»

Институт интеллектуальных кибернетических систем

Кафедра №12 «Компьютерные системы и технологии»



ОТЧЁТ

О выполнении лабораторной работы № 5

"Работа с массивами структур. Исследование методов сортировки массивов."

Студент: Гаврилов Д. А.

Группа: Б23-516

Преподаватель: Бабалова И. Ф.

Москва – 2023

1. Формулировка задания

Вариант №262

Введение

В ходе выполнения лабораторной работы должны быть разработаны:

1. Программа № 1, осуществляющая в диалоговом режиме ввод, сортировку и вывод последовательности данных, которая представляется в виде массива структур.
2. Программа № 2, осуществляющая таймирование сортировки массивов.

Программы № 1 и № 2 должны реализовывать поддержку работы с тремя алгоритмами сортировок: с двумя из состава индивидуального задания и с реализацией алгоритма быстрой сортировки из состава стандартной библиотеки — функцией `qsort()`.

Программа № 1 должна реализовывать следующую функциональность:

1. Ввод массива:

- (a) из стандартного потока ввода потока («с клавиатуры»);
- (b) из текстового файла (с возможностью указания имени файла);
- (c) из бинарного файла (с возможностью указания имени файла).

2. Вывод массива:

- (a) в стандартный поток вывода («на экран»);
- (b) в текстовый файл (с возможностью указания имени файла);
- (c) в бинарный файл (с возможностью указания имени файла).

3. Сортировка массива с возможностью выбора пользователем через диалоговое меню:

- (a) алгоритма сортировки (одного из трёх);
- (b) поля структуры, по которому осуществляется сортировка;
- (c) направления сортировки (по убыванию или по возрастанию).

Программа № 2 должна реализовывать таймирование сортировки с возможностью выбора пользователем через диалоговое меню:

1. алгоритма сортировки (одного из трёх);
2. поля структуры, по которому осуществляется сортировка;
3. направления сортировки (по убыванию или по возрастанию);
4. количества элементов в генерируемых массивах;
5. количества генерируемых массивов.

Примечания:

1. Взаимодействие программ с пользователем должно быть выстроено с помощью иерархического диалогового меню.
2. Программа № 1 должна осуществлять проверку корректности данных, вводимых пользователем, и, в случае ошибок, выдавать соответствующие сообщения, после чего продолжать работу.
3. Программа № 1 должна осуществлять проверку корректности данных, считываемых из файлов. В случае ошибок формата файла — выдавать соответствующие сообщения в стандартный поток вывода ошибок и продолжать работу, считая что ввод не был выполнен успешно. В случае некорректных данных для конкретных записей — выдавать соответствующие сообщения в поток ошибок, после чего продолжать работу, игнорируя данные записи.
4. Для работы с данными, формат которых описан в индивидуальном задании, должен быть разработан собственный составной тип данных — структура.
5. Для работы с данными, структура которых описана в индивидуальном задании, должен быть разработан формат хранения в текстовом файле.
6. Для работы с данными, структура которых описана в индивидуальном задании, должен быть разработан формат хранения в бинарном файле.
7. Работа с текстовыми файлами должна осуществляться при помощи функций стандартной библиотеки fopen(), fclose(), fprintf(), fscanf().
8. Работа с бинарными файлами должна осуществляться при помощи функций стандартной библиотеки fopen(), fclose(), fread(), fwrite().

9. Логически законченные части алгоритма решения задачи должны быть оформлены в виде отдельных функций с параметрами. Использование глобальных переменных не допускается.
10. Исходные коды программы должны быть логичным образом разбиты на несколько файлов.
11. Программа должна корректным образом работать с памятью, для проверки необходимо использовать соответствующие программные средства, например: valgrind (при тестировании и отладке программы ее необходимо запускать командой вида valgrind ./Lab5, а при анализе производительности — ./Lab5).

Отчётность по выполнению лабораторной работы должна включать:

1. Блок-схемы алгоритмов сортировки массива.
2. Исходные коды программ.
3. Тестовые наборы для иллюстрации работы программ.
4. Результаты таймирования, содержащие таблицы, графики зависимости времени выполнения сортировок от количества сортируемых элементов и аргументированные выводы об оценке сложности рассмотренных алгоритмов сортировки и её совпадении с теоретическими ожиданиями.

Индивидуальное задание

Структура данных

Автомобиль:

- марка (строка длиной до 16 символов, которая может включать в себя только буквы, дефис и пробелы);
- ФИО владельца (строка произвольной длины);
- пробег (дробное число, соответствующее величине пробега в тыс. км).

Алгоритмы сортировки

1. Пузырьковая сортировка (Bubble sort).
2. Двухсторонняя сортировка выбором (Double selection sort).

2. Описание использованных типов данных

Целочисленный тип данных – int (спецификатор формата %d) для хранения целых чисел и длин динамических массивов/строк. Тип данных с плавающей точкой – float (спецификатор формата %f), нужный в соответствии с индивидуальным заданием для хранения величин. Символьный тип данных - char – для работы с отдельными символами. Указатель на char (char *) – для хранения и создания строк. Для работы с файлами использовался тип данных FILE *. Также использовались пользовательские типы данных, указанные в формулировке индивидуального задания (car).

3. Описание использованных алгоритмов

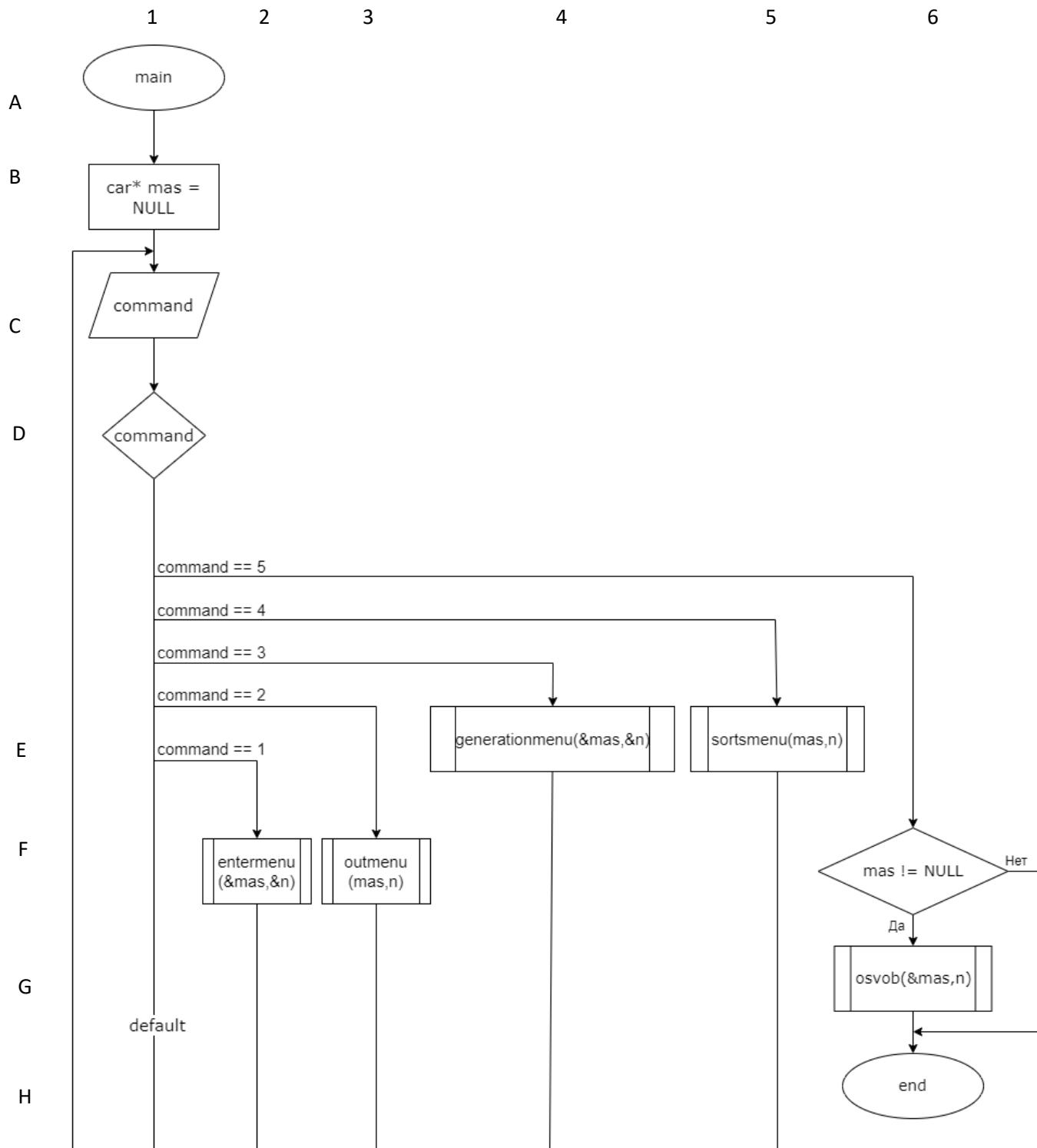


Рис. 1: Блок-схема алгоритма меню в главной функции `main()`.

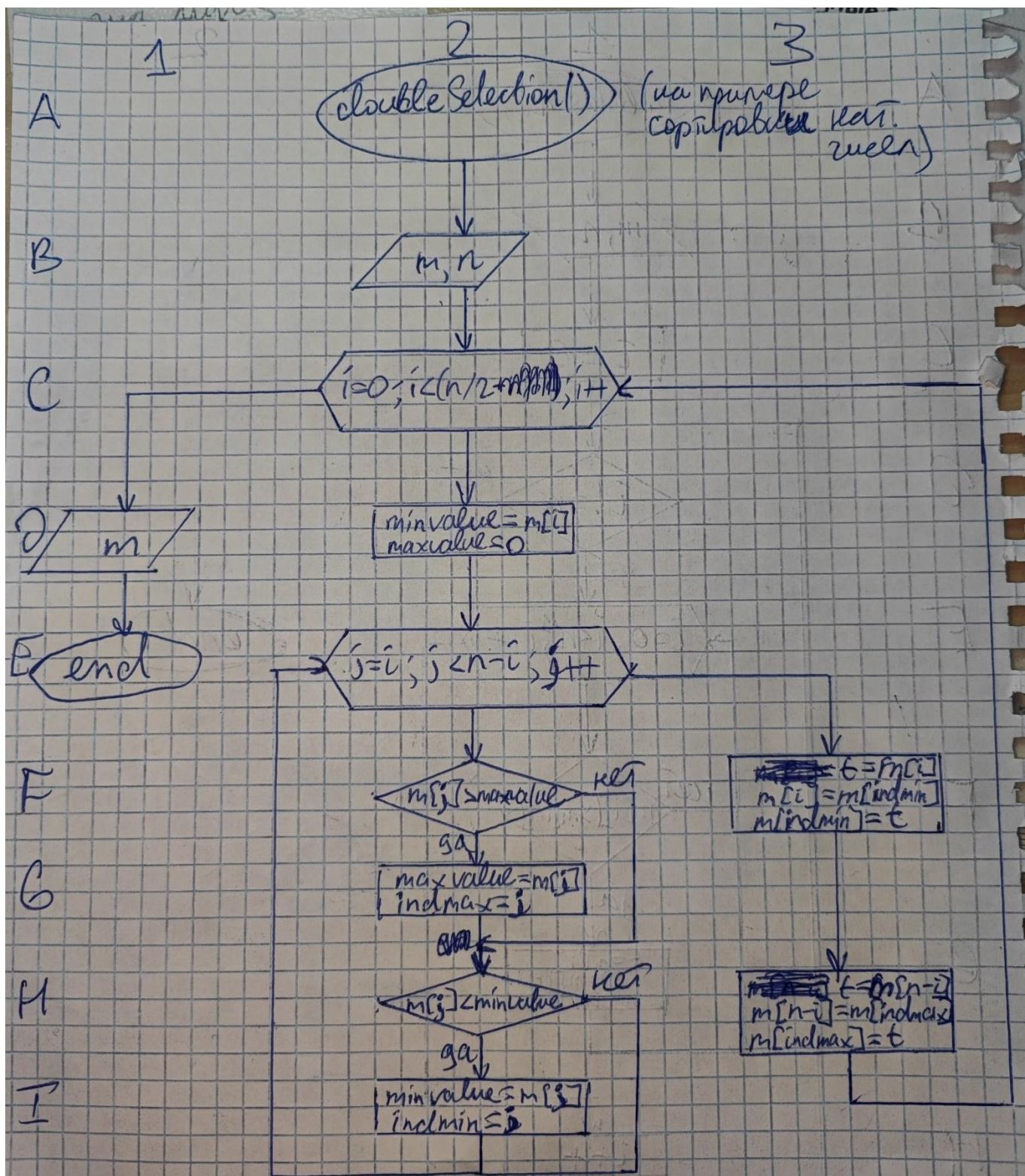


Рис. 2: Блок-схема алгоритма сортировки Double Selection на примере сортировки натуральных чисел.

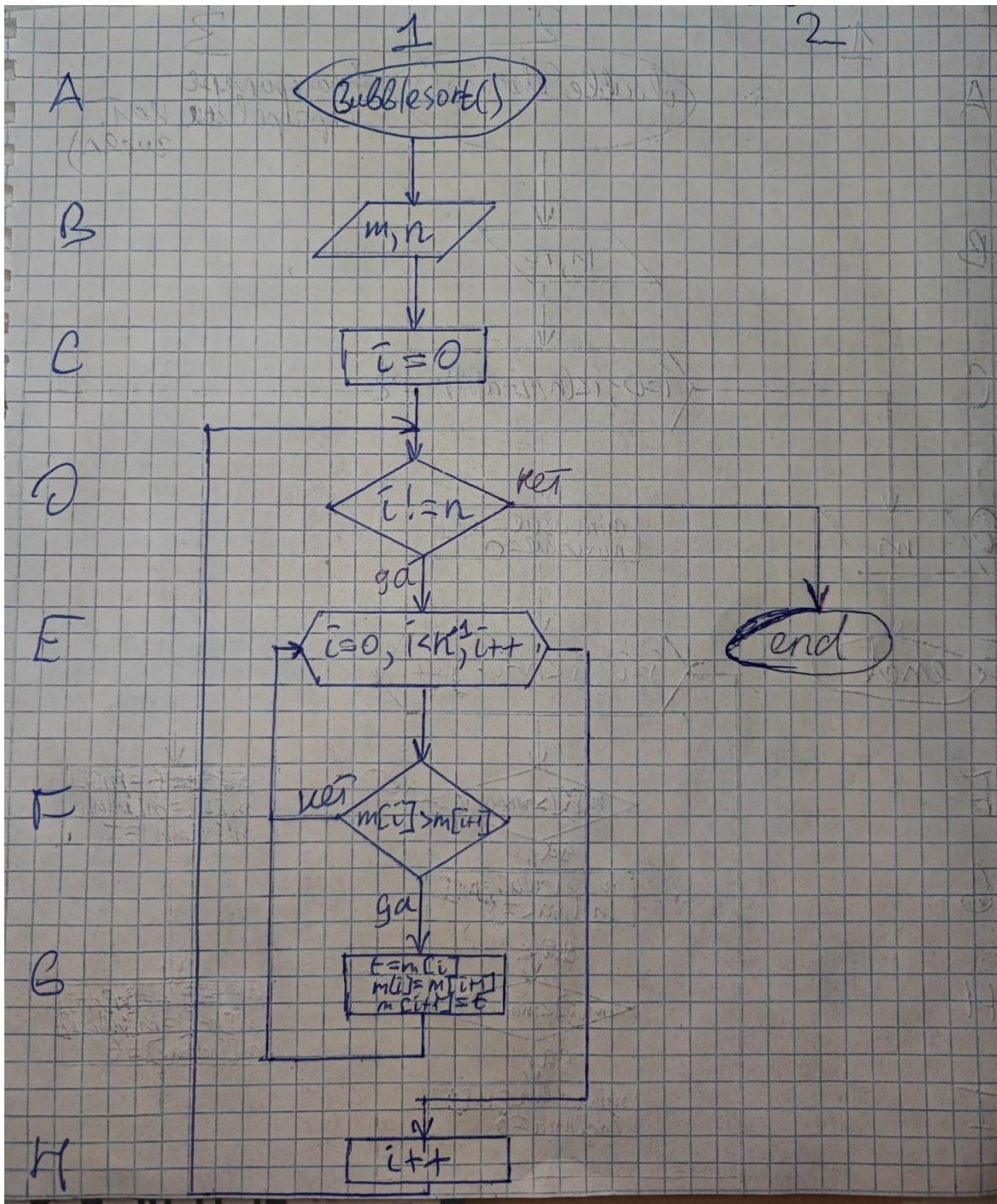


Рис. 3: Блок-схема алгоритма сортировки Bubble на примере сортировки чисел.

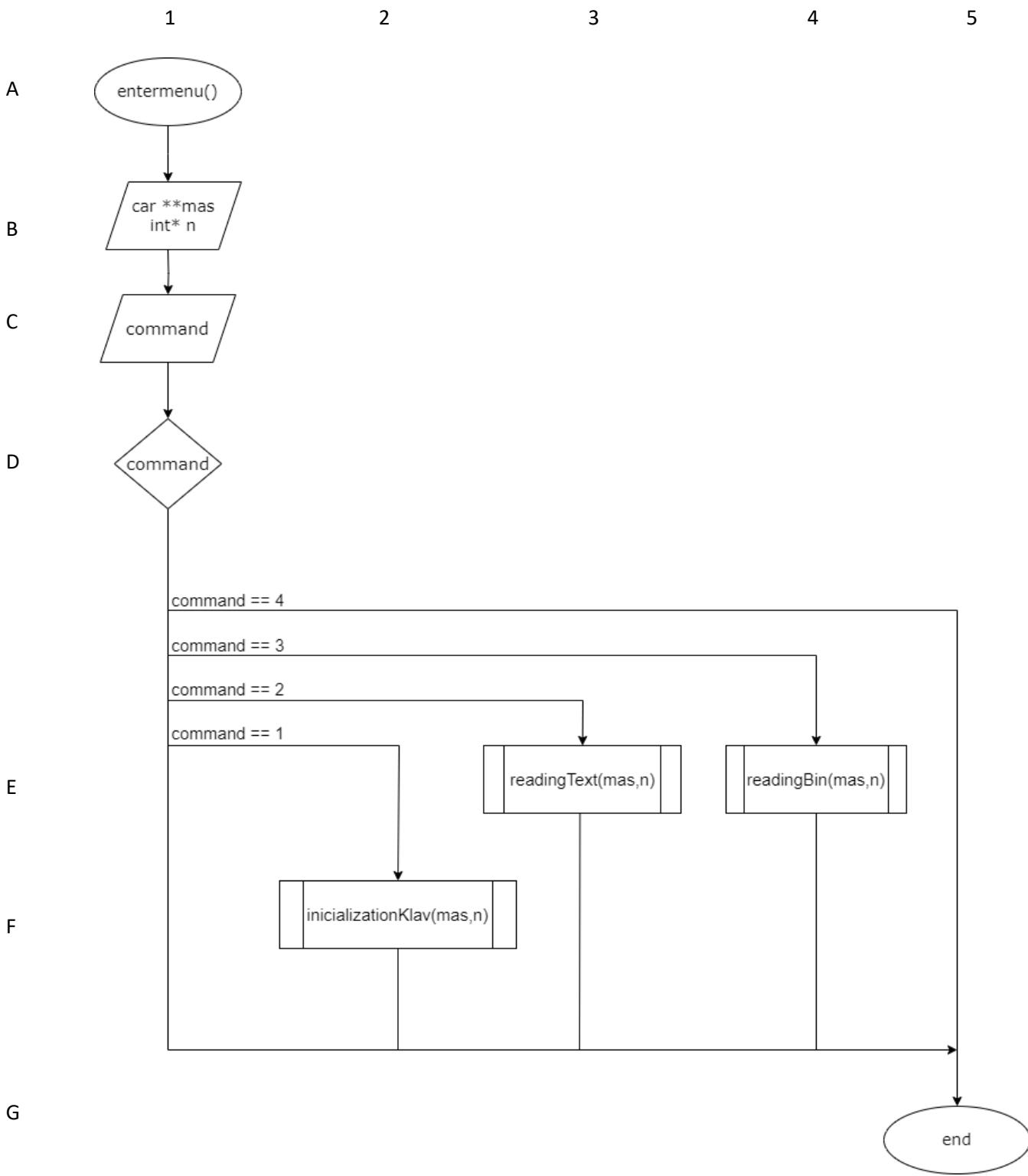


Рис. 4: Блок-схема алгоритма меню в функции ввода entermenu().

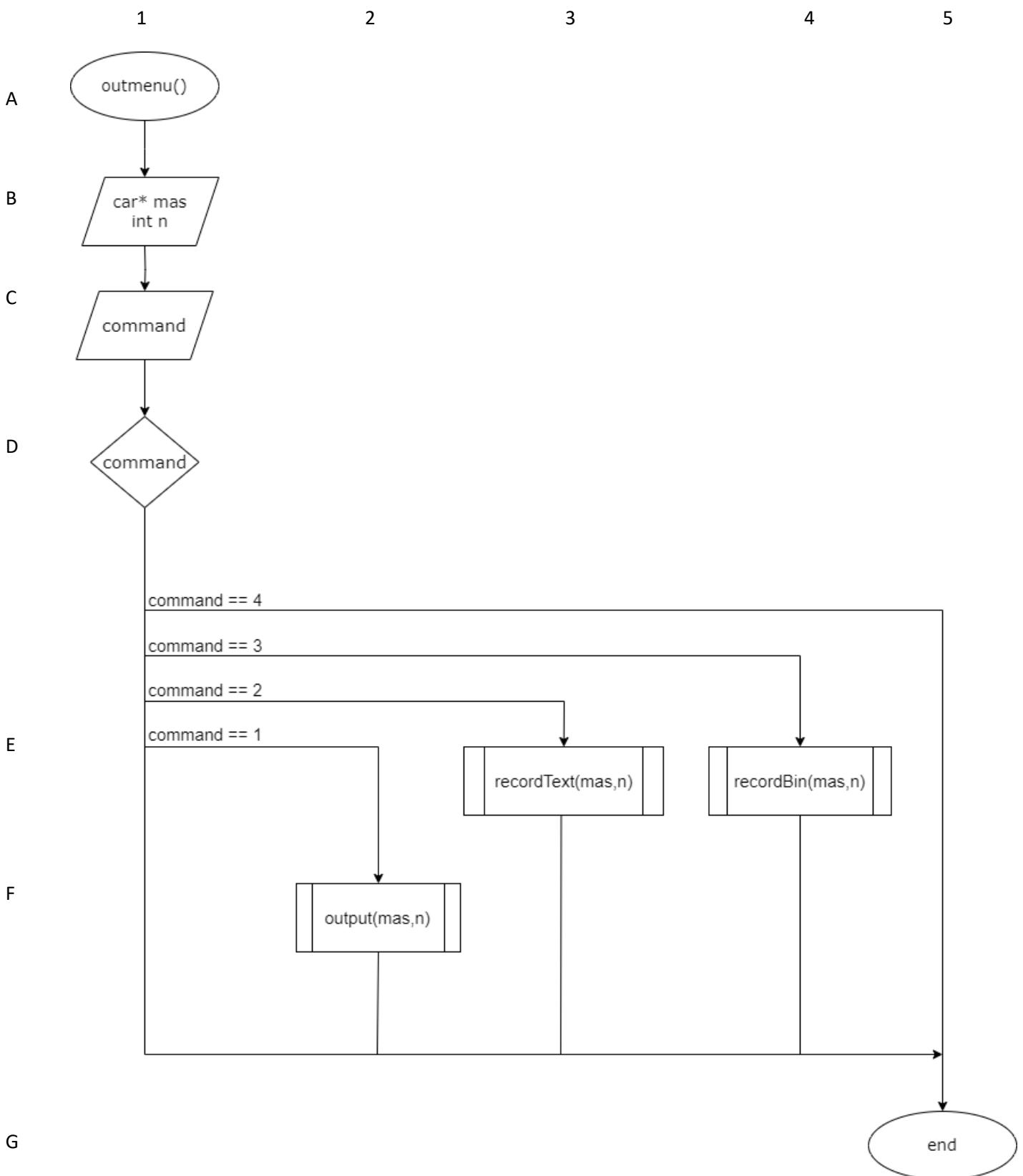


Рис. 5: Блок-схема алгоритма меню в функции вывода outmenu().

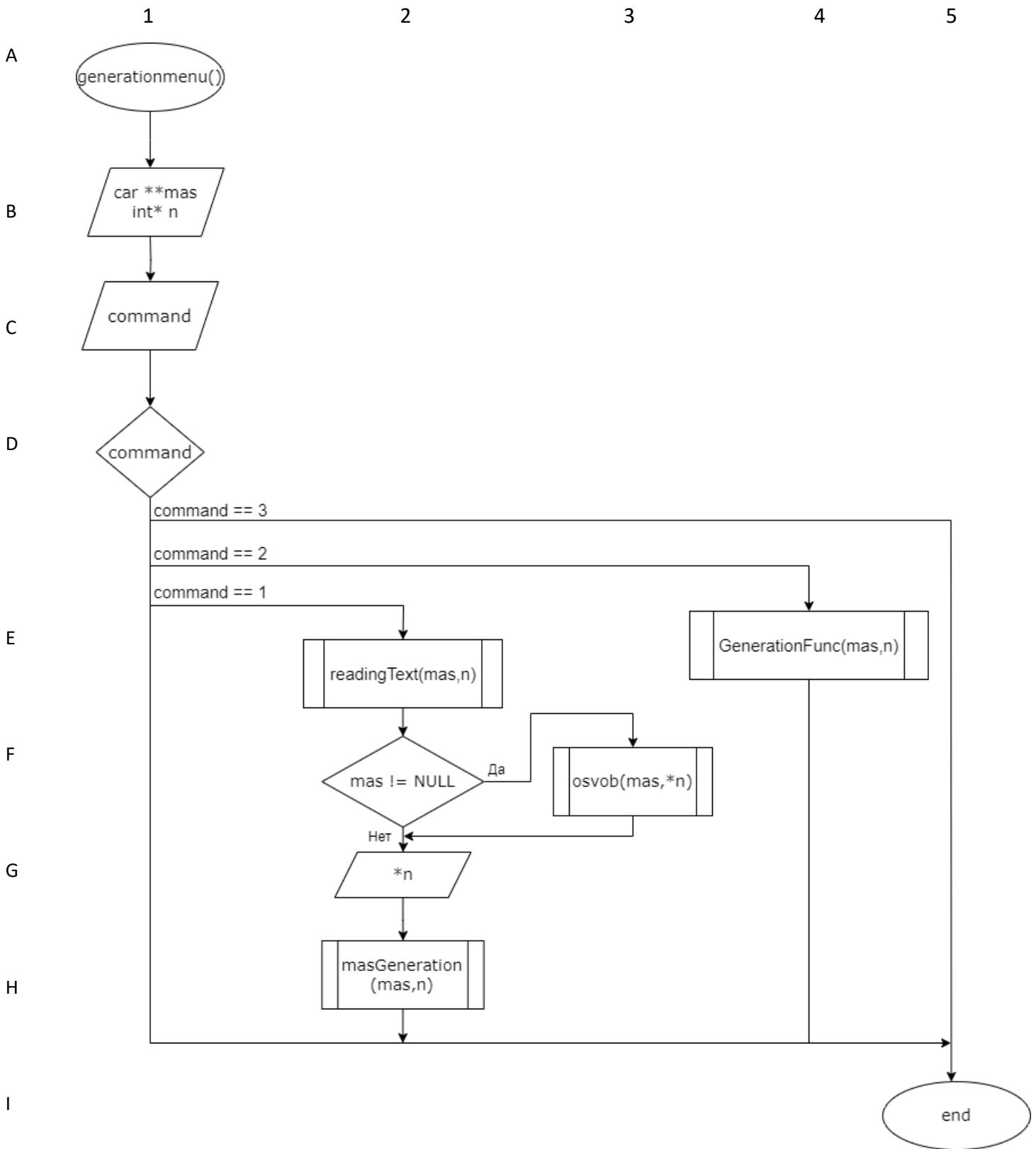


Рис. 6: Блок-схема алгоритма меню в функции генерации нового массива и таймирования generationmenu().

4. Исходные коды разработанных программ

Структура проекта состоит из 5 файлов: 1 FILELAB.h (файл прототипов) и 4 с исходным кодом – lab5.c, masfunctions.c, sorts.c, menu.c.

Листинг 1: Исходный код программы main (файл: lab5.c)

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4 #include <time.h>
5 #include "FILELAB.h"
6
7 int main(){
8     srand(time(NULL));
9     car* mas = NULL; int n; int command;
10    do{
11        printf("\nВведите команду:\n1) Ввод данных\n2) Вывод данных\n3) Генерация массива\n4) Сортировки\n5) Выход\n");
12        scanf("%d", &command); scanf("%*c");
13        switch(command){
14            case 1:
15                entermenu(&mas, &n);
16                break;
17            case 2:
18                outmenu(mas, n);
19                break;
20            case 3:
21                generationmenu(&mas, &n);
22                break;
23            case 4:
24                sortsmenu(mas, n);
25                break;
26            default:
27                if (command != 5){printf("Ошибка операции. Повторите ещё раз.\n");}
28                break;
29        }
30    }while(command != 5);
31    if (mas != NULL){osvob(&mas,n);}
32    return 0;
33 }
34 }
```

Листинг 2: Исходный код программы menu (файл: menu.c)

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4 #include <time.h>
5 #include "FILELAB.h"
6
7 void entermenu(car** mas, int* n){
8     printf("\n1) Ввод с клавиатуры\n2) Ввод из .txt\n3) Ввод из .bin\n4) Выход в главное меню\n");
9     int command;
10    scanf("%d", &command); scanf("%*c");
11    switch(command){
12        case 1:
13            initializationKlav(mas, n);
14            return;
15        case 2:
16            readingText(mas, n);
17            return;
18        case 3:
19            readingBin(mas, n);
20            return;
21        default:
22            return;
23    }
24 }
25
26 void outmenu(car* mas, int n){
27     printf("\n1) Вывод на экран\n2) Запись в .txt\n3) Запись в .bin\n4) Выход в главное меню\n");
28     int command;
29     scanf("%d", &command); scanf("%*c");
30     switch(command){
31         case 1:
32             output(mas, n);
33             return;
34         case 2:
35             recordText(mas, n);
36             return;
37         case 3:
38             recordBin(mas, n);
39             return;
40         default:
41             return;
42     }
43 }
44
45 void generationmenu(car** mas, int* n){
46     printf("\n1) Генерация массива структур заданной длины (1 массив)\n2) Исследование сортировки (программа 2)\n3) Выход в главное меню\n");
47     int command;
48     scanf("%d", &command); scanf("%*c");
49     switch(command){
50         case 1:
51             if (*mas != NULL){osvob(mas,*n);}
52             printf("Введите количество элементов массива структур: ");
53             scanf("%d", n); scanf("%*c");
54             masGeneration(mas, n);
55             printf("Массив из %d элементов успешно сгенерирован!\n", *n);
56             return;
57         case 2:
58             GenerationFunc(mas, n);
59             return;
60         default:
61             return;
62     }
63 }
```

```

65 void sortsmenu(car* mas, int n){
66     printf("\n1)qsort\n2)bubbleSort\n3)doubleSelectionSort\n4)Выход в главное меню\n");
67     int a; int b; int c;
68     scanf("%d", &a); scanf("%*c");
69     if ((a == 1) || (a == 2) || (a == 3)){
70         printf("\nВыберите поле:\n1 - марка\n2 - ФИО\n3 - пробег\n");
71         scanf("%d", &b); scanf("%*c");
72         printf("\nВыберите направление:\n1 - по возрастанию\n2 - по убыванию\n");
73         scanf("%d", &c); scanf("%*c");
74     }
75     switch(a){
76         case 1:
77             if(b == 1){
78                 if (c == 1){qsort(mas, n, sizeof(car), compMarkVozr);}
79                 else if (c == 2){qsort(mas, n, sizeof(car), compMarkUbiv);}
80             }
81             else if (b == 2){
82                 if (c == 1){qsort(mas, n, sizeof(car), compFioVozr);}
83                 else if (c == 2){qsort(mas, n, sizeof(car), compFioUbiv);}
84             }
85             else if (b == 3){
86                 if (c == 1){qsort(mas, n, sizeof(car), compProbegVozr);}
87                 else if (c == 2){qsort(mas, n, sizeof(car), compProbegUbiv);}
88             }
89             return;
90         case 2:
91             if(b == 1){
92                 if (c == 1){bubbleMark(mas, n, 0);}
93                 else if (c == 2){bubbleMark(mas, n, 1);}
94             }
95             else if (b == 2){
96                 if (c == 1){bubbleFio(mas, n, 0);}
97                 else if (c == 2){bubbleFio(mas, n, 1);}
98             }
99             else if (b == 3){
100                 if (c == 1){bubbleProbeg(mas, n, 0);}
101                 else if (c == 2){bubbleProbeg(mas, n, 1);}
102             }
103             return;
104
105         case 3:
106             if(b == 1){
107                 if (c == 1){doubleSelectionMark(mas, n, 0);}
108                 else if (c == 2){doubleSelectionMark(mas, n, 1);}
109             }
110             else if (b == 2){
111                 if (c == 1){doubleSelectionFio(mas, n, 0);}
112                 else if (c == 2){doubleSelectionFio(mas, n, 1);}
113             }
114             else if (b == 3){
115                 if (c == 1){doubleSelectionProbeg(mas, n, 0);}
116                 else if (c == 2){doubleSelectionProbeg(mas, n, 1);}
117             }
118             return;
119         default:
120             return;
121     }
122 }
```

Листинг 3: Исходный код программы masfunctions (файл: masfunctions.c)

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4 #include <time.h>
5 #include "FILELAB.h"
6
7 char* myreadline(){
8     char *ptr = (char*)malloc(1);
9     char buf[81];
10    int n, len = 0;
11    *ptr = '\0';
12    do {
13        n = scanf("%80[^\\n]",buf);
14        if (n < 0){
15            free(ptr);
16            ptr = NULL;
17            continue;
18        }
19        if (n == 0){scanf("%*c");}
20        else {
21            len += strlen(buf);
22            ptr = (char*) realloc(ptr,len + 1);
23            strcat(ptr,buf);
24        }
25    } while(n > 0);
26    return ptr;
27 }
28
29 char* myfilereadline(FILE* file){
30     char *ptr = (char*)malloc(1);
31     char buf[81];
32     int n, len = 0;
33     *ptr = '\0';
34     do {
35         n = fscanf(file,"%80[^\\n]",buf);
36         if (n < 0){
37             free(ptr);
38             ptr = NULL;
39             continue;
40         }
41         if (n == 0){fscanf(file,"%*c");}
42         else {
43             len += strlen(buf);
44             ptr = (char*) realloc(ptr,len + 1);
45             strcat(ptr,buf);
46         }
47     } while(n > 0);
48     return ptr;
49 }
50
51 void swap(car* car1, car* car2){
52     car t;
53     memcpy(&t,car1,sizeof(car));
54     memcpy(car1,car2,sizeof(car));
55     memcpy(car2,&t,sizeof(car));
56 }
```

```

57 int vvod_car_klav(car* mashina){//память на структуру выделили до, выводит 0 или 1 в зависимости от успешности ввода
58     printf("Введите марку: ");
59     scanf("%16[^\\n]", (mashina -> mark));// проверка на остаток символов, проверка на соответствие тому что в задании
60     (mashina -> mark)[16] = '\\0';
61     if (scanf("%16[^\\n]",(mashina -> mark)) == 1){scanf("%*[^\n]");scanf("%*c"); printf("Марка больше 16 символов! Введите ещё раз.\n"); return 0;
62     for (int i = 0; i < strlen(mashina -> mark); i++){
63         if (((mashina -> mark)[i] <= 122 && (mashina -> mark)[i] >= 97) || ((mashina -> mark)[i] <= 90 && (mashina -> mark)[i] >= 65) || ((m
64             scanf("%*[^\n]");scanf("%*c"); printf("Недопустимые символы поля \"Марка\". Введите ещё раз.\n"); return 0;
65         }
66     }
67     scanf("%*c");
68     printf("Введите фио: ");
69     mashina -> fio = myreadline();
70     printf("Введите пробег: ");
71     scanf("%f", &(mashina -> probeg)); // проверка что не осталось больше элементов в сканфе, что число
72     scanf("%*[^\n]");
73     scanf("%*c");
74     return 1;
75 }
76 }

78 void initializationKlav(car** mas, int* n){ // в мейне car* mas; int n;
79     int r;
80     if (*mas != NULL){osvob(mas,*n);}
81     printf("Введите количество структур: ");
82     scanf("%d", n);
83     scanf("%*c");
84     *mas = calloc(*n, sizeof(car));
85     for (int i = 0; i < *n; i++){
86         do{
87             r = vvod_car_klav(&((*mas)[i]));
88         }while(r == 0);
89     }
90 }

92 void output(car* mas, int n){
93     if (mas == NULL){printf("Массив не введён!\n"); return;}
94     printf("Выход массива структур:\n");
95     for (int i = 0; i < n; i++){
96         printf("Марка: \"%s\" ФИО: \"%s\" Пробег: \"%f\"\n", mas[i].mark, mas[i].fio, mas[i].probeg);
97     }
98     printf("Массив выведен.\n\n");
99 }

100 void osvob(car** mas, int n){
101     for (int i = 0; i < n; i++){
102         free((*mas)[i].fio);
103     }
104     free(*mas);
105     *mas = NULL;
106 }
107 }

109 void recordText(car* mas, int n){
110     if (mas == NULL){printf("Массив не введён!\n"); return;}
111     printf("Введите имя текстового файла (+.txt): ");
112     char * filename = myreadline();
113     FILE* file = fopen(filename,"w+t");
114     for (int i = 0; i < n; i++){
115         fprintf(file,"%s\n%s\n%f\n", mas[i].mark, mas[i].fio, mas[i].probeg);
116     }
117     fclose(file);
118     free(filename);
119 }

121 void readingText(car** mas, int* n){
122     printf("Введите имя текстового файла (+.txt): ");
123     char * filename = myreadline();
124     FILE* file = fopen(filename,"r+t");
125     if(file != NULL){
126         if (*mas != NULL){osvob(mas,*n);}
127         *n = 0;
128         int i = 0;
129         *mas = (car*)calloc(1,sizeof(car));
130         while (fscanf(file,"%16[^\\n]", (*mas)[i].mark) != EOF){
131             (*mas)[i].mark[16] = '\\0';
132             fscanf(file,"%*c");
133             (*mas)[i].fio = myfilereadline(file);
134             fscanf(file, "%f", &((*mas)[i].probeg));
135             fscanf(file,"%*c");
136             *n += 1;
137             i++;
138             *mas = (car*)realloc(*mas, (*n+1) * sizeof(car));
139         }
140         *mas = (car*)realloc(*mas, (*n) * sizeof(car));
141         fclose(file);
142         free(filename);
143     }
144     else{
145         printf("Ошибка чтения файла .txt\n");
146         fclose(file);
147         if (filename != NULL){free(filename);}
148     }
149 }

```

```
151 void recordBin(car* mas, int n){
152     if (mas == NULL){printf("Массив не введён!\n"); return;}
153     printf("Введите имя бинарного файла (+.bin): ");
154     char * filename = myreadline();
155     FILE* file = fopen(filename,"w+b");
156     int k;
157     for (int i = 0; i < n; i++){
158         k = strlen(mas[i].mark);
159         fwrite(&k, sizeof(int), 1, file);
160         fwrite(mas[i].mark, sizeof(char), k, file);
161         k = strlen(mas[i].fio);
162         fwrite(&k, sizeof(int), 1, file);
163         fwrite(mas[i].fio, sizeof(char), k, file);
164         fwrite(&(mas[i].probeg), sizeof(float), 1, file);
165     }
166     fclose(file);
167     free(filename);
168 }
```

```
170 void readingBin(car** mas, int* n){
171     printf("Введите имя текстового файла (+.bin): ");
172     char * filename = myreadline();
173     FILE* file = fopen(filename,"r+b");
174     if(file != NULL){
175         if (*mas != NULL){osvob(mas,*n);}
176         *n = 0;
177         int i = 0;
178         int size;
179         *mas = (car*)calloc(1, sizeof(car));
180         while (fread(&size, sizeof(int), 1, file) == 1){
181             fread((*mas)[i].mark, sizeof(char), size, file);
182             (*mas)[i].mark[size] = '\0';
183             fread(&size, sizeof(int), 1, file);
184             (*mas)[i].fio = calloc(size+1, sizeof(char));
185             fread((*mas)[i].fio, sizeof(char), size, file);
186             (*mas)[i].fio[size] = '\0';
187             fread(&(*mas)[i].probeg, sizeof(float), 1, file);
188             *n += 1;
189             i++;
190             *mas = (car*)realloc(*mas, (*n+1) * sizeof(car));
191         }
192         *mas = (car*)realloc(*mas, (*n) * sizeof(car));
193         fclose(file);
194         free(filename);
195     }
196     else{
197         printf("Ошибка чтения файла .bin\n");
198         fclose(file);
199         if (filename != NULL){free(filename);}
200     }
201 }
```

```

203 void masGeneration(car** mas, int* n){ // n - кол во в массиве, перед отдельным mas generation надо очищать
204     int randomlen;
205     char randomletter;
206     *mas = calloc(*n, sizeof(car));
207     for (int i = 0; i < *n; i++){
208         randomlen = rand()%16+1;
209         for(int j = 0; j < randomlen; j++){
210             randomletter = rand()%26+97;
211             ((*mas)[i].mark)[j] = randomletter;
212         }
213         ((*mas)[i].mark)[randomlen] = '\0';
214         randomlen = rand()%20+10;
215         (*mas)[i].fio = calloc(randomlen+1, sizeof(char));
216         for(int j = 0; j < randomlen; j++){
217             randomletter = rand()%26+97;
218             ((*mas)[i].fio)[j] = randomletter;
219         }
220         ((*mas)[i].fio)[randomlen] = '\0';
221         (*mas)[i].probeg = (float)(rand()%500000+1000);
222     }
223 }
224

```

```

225 void GenerationFunc(car** mas, int* n){ // исследование сортировки (программа 2)
226     clock_t summary; int m; int k; int a; int b; int c;
227     clock_t end; clock_t start;
228     printf("Выберите сортировку:\n1 - qsort\n2 - bubbleSort\n3 - doubleSelectionSort\n");
229     scanf("%d", &a); scanf("%*c");
230     printf("Выберите поле:\n1 - марка\n2 - ФИО\n3 - пробег\n");
231     scanf("%d", &b); scanf("%*c");
232     printf("Выберите направление:\n1 - по возрастанию\n2 - по убыванию\n");
233     scanf("%d", &c); scanf("%*c");
234     printf("Введите количество генераций массива: ");
235     scanf("%d", &m); scanf("%*c");
236     printf("Введите количество элементов в генерируемом массиве: ");
237     scanf("%d", &k); scanf("%*c");
238     if(*mas != NULL){osvob(mas, *n);}
239     *n = k;
240     for (int i = 0; i < m; i++){
241         if(*mas != NULL){osvob(mas, *n);}
242         masGeneration(mas,n);
243         start = clock();
244         if (a == 1){
245             if(b == 1){
246                 if (c == 1){qsort(*mas, *n, sizeof(car), compMarkVozr);}
247                 else if (c == 2){qsort(*mas, *n, sizeof(car), compMarkUbiv);}
248             }
249             else if (b == 2){
250                 if (c == 1){qsort(*mas, *n, sizeof(car), compFioVozr);}
251                 else if (c == 2){qsort(*mas, *n, sizeof(car), compFioUbiv);}
252             }
253             else if (b == 3){
254                 if (c == 1){qsort(*mas, *n, sizeof(car), compProbegVozr);}
255                 else if (c == 2){qsort(*mas, *n, sizeof(car), compProbegUbiv);}
256             }
257         }

```

```

258     }
259     else if(a == 2){
260         if(b == 1){
261             if (c == 1){bubbleMark(*mas, *n, 0);}
262             else if (c == 2){bubbleMark(*mas, *n, 1);}
263         }
264         else if (b == 2){
265             if (c == 1){bubbleFio(*mas, *n, 0);}
266             else if (c == 2){bubbleFio(*mas, *n, 1);}
267         }
268         else if (b == 3){
269             if (c == 1){bubbleProbeg(*mas, *n, 0);}
270             else if (c == 2){bubbleProbeg(*mas, *n, 1);}
271         }
272     }
273     else if(a == 3){
274         if(b == 1){
275             if (c == 1){doubleSelectionMark(*mas, *n, 0);}
276             else if (c == 2){doubleSelectionMark(*mas, *n, 1);}
277         }
278         else if (b == 2){
279             if (c == 1){doubleSelectionFio(*mas, *n, 0);}
280             else if (c == 2){doubleSelectionFio(*mas, *n, 1);}
281         }
282         else if (b == 3){
283             if (c == 1){doubleSelectionProbeg(*mas, *n, 0);}
284             else if (c == 2){doubleSelectionProbeg(*mas, *n, 1);}
285         }
286     }
287     end = clock();
288     summary += (end - start);
289 }
290 osvob(mas,*n);
291 printf("Среднее время сортировки одного массива (на входе %d массивов из %d элементов) - %lf c\n", m, k, ((double)summary) / m / CLOCKS_PER_SEC);

```

Листинг 4: Исходный код программы sorts (файл: sorts.c)

```

1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4 #include <time.h>
5 #include "FILELAB.h"
6
7 void doubleSelectionProbeg(car* mas, int n, int code){ // code == 0 - по возрастанию, code == 1 - по убыванию
8     if (mas == NULL){printf("Массив не введён!\n"); return;}
9     int indmax;int indmin;
10    float minvalue;float maxvalue;
11    clock_t start = clock();
12    for (int i = 0; i < (n/2); i++){
13        minvalue = mas[i].probeg;
14        maxvalue = mas[i].probeg;
15        indmax = i; indmin = i;
16        for (int j = i; j < n - i; j++){
17            if (mas[j].probeg > maxvalue){
18                maxvalue = mas[j].probeg;
19                indmax = j;
20            }
21            if (mas[j].probeg < minvalue){
22                minvalue = mas[j].probeg;
23                indmin = j;
24            }
25        }
26        if (code == 0){
27            if ((i == indmax) && ((n-i-1) == indmin)){swap(&(mas[i]),&(mas[n-i-1]));}// максимум в начале, мин в конце
28            else if((i == indmax) && ((n-i-1 != indmin))){swap(&(mas[n-i-1]), &(mas[indmax])); swap(&(mas[i]), &(mas[indmin]));}
29            else{
30                if (indmin != i){swap(&(mas[i]), &(mas[indmin]));}
31                if (indmax != n-i-1){swap(&(mas[n-i-1]), &(mas[indmax]));}
32            }
33        }
34        else if (code == 1){
35            if ((indmax == n-i-1) && (indmin == i)){swap(&(mas[i]), &(mas[n-i-1]));} // максимум в начале, мин в конце
36            else if ((indmax == n-i-1) && (indmin != i)){swap(&(mas[i]), &(mas[indmax]));swap(&(mas[n-i-1]), &(mas[indmin]));}
37            else{
38                if (indmin != n-i-1){swap(&(mas[n-i-1]), &(mas[indmin]));}
39                if (indmax != i){swap(&(mas[i]), &(mas[indmax]));}
40            }
41        }
42    }
43    clock_t end = clock();

```

```

43     clock_t end = clock();
44     printf("Сортировка заняла %lf с\n", (double)(end - start)/CLOCKS_PER_SEC);
45 }
46
47 void doubleSelectionMark(car* mas, int n, int code){ // code == 0 - по возрастанию, code == 1 - по убыванию
48     if (mas == NULL){printf("Массив не введен!\n"); return;}
49     char* minvalue;
50     char* maxvalue;
51     int indmax;
52     int indmin;
53     clock_t start = clock();
54     for (int i = 0; i < (n/2); i++){
55         minvalue = calloc(17, sizeof(char));
56         maxvalue = calloc(17, sizeof(char));
57         strcpy(minvalue, mas[i].mark);
58         strcpy(maxvalue, mas[i].mark);
59         indmax = i; indmin = i;
60         for (int j = i; j < n - i; j++){
61             if ((strcmp(mas[j].mark, maxvalue)) > 0){
62                 free(maxvalue);
63                 maxvalue = calloc(17, sizeof(char));
64                 strcpy(maxvalue, mas[j].mark);
65                 indmax = j;
66             }
67             if ((strcmp(mas[j].mark, minvalue)) < 0){
68                 free(minvalue);
69                 minvalue = calloc(17, sizeof(char));
70                 strcpy(minvalue, mas[j].mark);
71                 indmin = j;
72             }
73         }
74         if (code == 0){
75             if ((i == indmax) && ((n-i-1) == indmin)){swap(&(mas[i]), &(mas[n-i-1]));} // максимум в начале, мин в конце
76             else if((i == indmax) && ((n-i-1 != indmin))){swap(&(mas[n-i-1]), &(mas[indmax])); swap(&(mas[i]), &(mas[indmin]));}
77             else{
78                 if (indmin != i){swap(&(mas[i]), &(mas[indmin]));}
79                 if (indmax != n-i-1){swap(&(mas[n-i-1]), &(mas[indmax]));}
80             }
81         }
82         else if (code == 1){
83             if ((indmax == n-i-1) && (indmin == i)){swap(&(mas[i]), &(mas[n-i-1]));} // максимум в начале, мин в конце

```

```

83
84         if ((indmax == n-i-1) && (indmin == i)){swap(&(mas[i]), &(mas[n-i-1]));} // максимум в начале, мин в конце
85         else if ((indmax == n-i-1) && (indmin != i)){swap(&(mas[i]), &(mas[indmax]));swap(&(mas[n-i-1]), &(mas[indmin]));}
86             else{
87                 if (indmin != n-i-1){swap(&(mas[n-i-1]), &(mas[indmin]));}
88                 if (indmax != i){swap(&(mas[i]), &(mas[indmax]));}
89             }
90         free(minvalue);
91         free(maxvalue);
92     }
93     clock_t end = clock();
94     printf("Сортировка заняла %lf с\n", (double)(end - start)/CLOCKS_PER_SEC);
95 }
96
97 void doubleSelectionFio(car* mas, int n, int code){ // code == 0 - по возрастанию, code == 1 - по убыванию
98     if (mas == NULL){printf("Массив не введен!\n"); return;}
99     char* minvalue;
100    char* maxvalue;
101    int indmax;
102    int indmin;
103    clock_t start = clock();
104    for (int i = 0; i < (n/2); i++){
105        minvalue = calloc(strlen(mas[i].fio)+1, sizeof(char));
106        maxvalue = calloc(strlen(mas[i].fio)+1, sizeof(char));
107        strcpy(minvalue, mas[i].fio);
108        strcpy(maxvalue, mas[i].fio);
109        indmax = i; indmin = i;
110        for (int j = i; j < n - i; j++){
111            if ((strcmp(mas[j].fio, maxvalue)) > 0){
112                free(maxvalue);
113                maxvalue = calloc(strlen(mas[j].fio)+1, sizeof(char));
114                strcpy(maxvalue, mas[j].fio);
115                indmax = j;
116            }
117            if ((strcmp(mas[j].fio, minvalue)) < 0){
118                free(minvalue);
119                minvalue = calloc(strlen(mas[j].fio)+1, sizeof(char));
120                strcpy(minvalue, mas[j].fio);
121                indmin = j;
122            }
123        }

```

```

124     if (code == 0){
125         if ((i == indmax) && ((n-i-1) == indmin)){swap(&(mas[i]),&(mas[n-i-1]));}// максимум в начале, мин в конце
126         else if((i == indmax) && ((n-i-1 != indmin))){swap(&(mas[n-i-1]), &(mas[indmax])); swap(&(mas[i]), &(mas[indmin]));}
127         else{
128             if (indmin != i){swap(&(mas[i]), &(mas[indmin]));}
129             if (indmax != n-i-1){swap(&(mas[n-i-1]), &(mas[indmax]));}
130         }
131     }
132     else if (code == 1){
133         if ((indmax == n-i-1) && (indmin == i)){swap(&(mas[i]), &(mas[n-i-1]));} // максимум в начале, мин в конце
134         else if ((indmax == n-i-1) && (indmin != i)){swap(&(mas[i]), &(mas[indmax]));swap(&(mas[n-i-1]), &(mas[indmin]));} //
135         else{
136             if (indmin != n-i-1){swap(&(mas[n-i-1]), &(mas[indmin]));}
137             if (indmax != i){swap(&(mas[i]), &(mas[indmax]));}
138         }
139     }
140     free(minvalue);
141     free(maxvalue);
142 }
143 clock_t end = clock();
144 printf("Сортировка заняла %lf с\n", (double)(end - start)/CLOCKS_PER_SEC);
145 }

146 void bubbleProbeg(car* mas, int n, int code){ // code == 0 - по возрастанию, code == 1 - по убыванию
147     if (mas == NULL){printf("Массив не введен!\n"); return;}
148     clock_t start = clock();
149     for(int i = 0; i < n-1; i++){
150         for(int j = 0; j < n-i-1; j++){
151             if (code == 0){
152                 if (mas[j].probeg > mas[j+1].probeg){
153                     swap(&mas[j],&mas[j+1]);
154                 }
155             }
156             if (code == 1){
157                 if (mas[j].probeg < mas[j+1].probeg){
158                     swap(&mas[j],&mas[j+1]);
159                 }
160             }
161         }
162     }
163     clock_t end = clock();
164     printf("Сортировка заняла %lf с\n", (double)(end - start)/CLOCKS_PER_SEC);
165 }
166 }

167 void bubbleMark(car* mas, int n, int code){ // code == 0 - по возрастанию, code == 1 - по убыванию
168     if (mas == NULL){printf("Массив не введен!\n"); return;}
169     clock_t start = clock();
170     for(int i = 0; i < n-1; i++){
171         for(int j = 0; j < n-i-1; j++){
172             if (code == 0){
173                 if ((strcmp(mas[j].mark, mas[j+1].mark)) > 0){
174                     swap(&mas[j],&(mas[j+1]));
175                 }
176             }
177             if (code == 1){
178                 if ((strcmp(mas[j].mark, mas[j+1].mark)) < 0){
179                     swap(&mas[j],&(mas[j+1]));
180                 }
181             }
182         }
183     }
184     clock_t end = clock();
185     printf("Сортировка заняла %lf с\n", (double)(end - start)/CLOCKS_PER_SEC);
186 }
187 }

188 void bubbleFio(car* mas, int n, int code){ // code == 0 - по возрастанию, code == 1 - по убыванию
189     if (mas == NULL){printf("Массив не введен!\n"); return;}
190     clock_t start = clock();
191     for(int i = 0; i < n-1; i++){
192         for(int j = 0; j < n-i-1; j++){
193             if (code == 0){
194                 if ((strcmp(mas[j].fio, mas[j+1].fio)) > 0){
195                     swap(&mas[j],&mas[j+1]);
196                 }
197             }
198             if (code == 1){
199                 if ((strcmp(mas[j].fio, mas[j+1].fio)) < 0){
200                     swap(&mas[j],&mas[j+1]);
201                 }
202             }
203         }
204     }
205     clock_t end = clock();
206     printf("Сортировка заняла %lf с\n", (double)(end - start)/CLOCKS_PER_SEC);
207 }
208 
```

```

162     }
163     clock_t end = clock();
164     printf("Сортировка заняла %lf с\n", (double)(end - start)/CLOCKS_PER_SEC);
165 }
166 }

167 void bubbleMark(car* mas, int n, int code){ // code == 0 - по возрастанию, code == 1 - по убыванию
168     if (mas == NULL){printf("Массив не введен!\n"); return;}
169     clock_t start = clock();
170     for(int i = 0; i < n-1; i++){
171         for(int j = 0; j < n-i-1; j++){
172             if (code == 0){
173                 if ((strcmp(mas[j].mark, mas[j+1].mark)) > 0){
174                     swap(&mas[j],&(mas[j+1]));
175                 }
176             }
177             if (code == 1){
178                 if ((strcmp(mas[j].mark, mas[j+1].mark)) < 0){
179                     swap(&mas[j],&(mas[j+1]));
180                 }
181             }
182         }
183     }
184     clock_t end = clock();
185     printf("Сортировка заняла %lf с\n", (double)(end - start)/CLOCKS_PER_SEC);
186 }
187 }

188 void bubbleFio(car* mas, int n, int code){ // code == 0 - по возрастанию, code == 1 - по убыванию
189     if (mas == NULL){printf("Массив не введен!\n"); return;}
190     clock_t start = clock();
191     for(int i = 0; i < n-1; i++){
192         for(int j = 0; j < n-i-1; j++){
193             if (code == 0){
194                 if ((strcmp(mas[j].fio, mas[j+1].fio)) > 0){
195                     swap(&mas[j],&mas[j+1]);
196                 }
197             }
198             if (code == 1){
199                 if ((strcmp(mas[j].fio, mas[j+1].fio)) < 0){
200                     swap(&mas[j],&mas[j+1]);
201                 }
202             }
203         }
204     }
205     clock_t end = clock();
206     printf("Сортировка заняла %lf с\n", (double)(end - start)/CLOCKS_PER_SEC);
207 }
208 
```

```

202 }
203     }
204 }
205 }
206     clock_t end = clock();
207     printf("Сортировка заняла %lf с\n", (double)(end - start)/CLOCKS_PER_SEC);
208 }
209
210 int compProbegVozr(const void * a, const void * b){
211     const car* first = (const car*) a;
212     const car* second = (const car*) b;
213     return (first->probeg - second->probeg);
214 }
215
216 int compProbegUbiv(const void * a, const void * b){
217     const car* first = (const car*) a;
218     const car* second = (const car*) b;
219     return (second->probeg - first->probeg);
220 }
221
222 int compMarkVozr(const void * a, const void * b){
223     const car* first = (const car*) a;
224     const car* second = (const car*) b;
225     return (strcmp(first->mark,second->mark));
226 }
227
228 int compMarkUbiv(const void * a, const void * b){
229     const car* first = (const car*) a;
230     const car* second = (const car*) b;
231     return (-(strcmp(first->mark,second->mark)));
232 }
233
234 int compFioVozr(const void * a, const void * b){
235     const car* first = (const car*) a;
236     const car* second = (const car*) b;
237     return (strcmp(first->fio,second->fio));
238 }
239
240 int compFioUbiv(const void * a, const void * b){
241     const car* first = (const car*) a;
242         const car* second = (const car*) b;
243         return (-(strcmp(first->fio,second->fio)));
244 }

```

Листинг 5: Исходный код программы FILELAB (файл: FILELAB.h)

```

1 #ifndef FILELAB_H
2 #define FILELAB_H
3
4 typedef struct car { // наша структура
5     char mark[17];
6     char* fio;
7     float probeg;
8 } car;
9
10 void inicializationKlav(car** mas, int* n); // для работы с массивом
11 void masGeneration(car** mas, int* n);
12 void GenerationFunc(car** mas, int* n);
13 void output(car* mas, int n);
14 void osvob(car** mas, int n);
15 int vvod_car_klav(car* mashina);
16 char* myreadline();
17 char* myfilereadline(FILE* file);
18 void swap(car* car1, car* car2);
19
20 void recordText(car** mas, int n); // запись считывание из файлов
21 void readingText(car** mas, int* n);
22 void recordBin(car* mas, int n);
23 void readingBin(car** mas, int* n);
24
25 void entermenu(car** mas, int* n); // все меню
26 void outmenu(car* mas, int n);
27 void generationmenu(car** mas, int* n);
28 void sortsmenu(car* mas, int n);
29
30 void doubleSelectionProbeg(car* mas, int n, int code); // сортировки
31 void doubleSelectionMark(car* mas, int n, int code);
32 void doubleSelectionFio(car* mas, int n, int code);
33 void bubbleProbeg(car* mas, int n, int code);
34 void bubbleMark(car* mas, int n, int code);
35 void bubbleFio(car* mas, int n, int code);
36
37 int compProbegVozr(const void * a, const void * b); // компараторы (для ксорта)
38 int compProbegUbiv(const void * a, const void * b);
39 int compMarkVozr(const void * a, const void * b);
40 int compMarkUbiv(const void * a, const void * b);
41 int compFioVozr(const void * a, const void * b);
42 int compFioUbiv(const void * a, const void * b);
43 #endif

```

5. Текстовые примеры работы программы (далее будут приложены скриншоты работы программы с этими данными). Массивы для тестов генерируются случайно по количеству элементов в них (задаёт пользователь). Для генерации в программе предусмотрена собственная функция.

Входной массив	Поле сортировки	Направление сортировки (1 – по возрастанию 2 – по убыванию)	Выходной массив после сортировки
<p>Марка: "e" ФИО: "jcxurmzyysmvjydksmrzlnllnf" Пробег: "381219.000000" Марка: "toddssadlofjpxykj" ФИО: "ptqdmrpozcbmh" Пробег: "111973.000000" Марка: "db" ФИО: "gvjlhaqspqqbzorueugwjha" Пробег: "259934.000000" Марка: "kz" ФИО: "ncivyrgftzxip" Пробег: "232324.000000" qsort - сортировка</p>	марка	2	<p>Марка: "toddssadlofjpxykj" ФИО: "ptqdmrpozcbmh" Пробег: "111973.000000" Марка: "kz" ФИО: "ncivyrgftzxip" Пробег: "232324.000000" Марка: "e" ФИО: "jcxurmzyysmvjydksmrzlnllnf" Пробег: "381219.000000" Марка: "db" ФИО: "gvjlhaqspqqbzorueugwjha" Пробег: "259934.000000"</p>
<p>Марка: "qgdkn" ФИО: "slizlxsvwrkzzfxtoemnd" Пробег: "499079.000000" Марка: "tjwamgplyaw" ФИО: "ltrjrltqmbphptwex" Пробег: "364349.000000" Марка: "segusoj" ФИО: "ojbnxwgiorbggewpt" Пробег: "360527.000000"</p>	пробег	1	<p>Марка: "zqijlmrfgrmtcng" ФИО: "mnjcgnkorienarmsaydoplv" Пробег: "262667.000000" Марка: "froricmpdvpn" ФИО: "cyqbczdmuyuiuhxo" Пробег: "332750.000000" Марка: "segusoj" ФИО: "ojbnxwgiorbggewpt"</p>

<p>Марка: "zqijlmrgfrmtcng" ФИО: "mnjcggnkorienarmsaydoplv" Пробег: "262667.000000" Марка: "hqgwzikmrosgz" ФИО: "ujnljgzeizhxruuoydg" Пробег: "497127.000000" Марка: "froricmpdvpn" ФИО: "cyqbczdmuyuiuhxo" Пробег: "332750.000000" qsort - сортировка</p>			<p>Пробег: "360527.000000" Марка: "tjwamgplyaw" ФИО: "ltrjrltqmbphptwex" Пробег: "364349.000000" Марка: "hqgwzikmrosgz" ФИО: "ujnljgzeizhxruuoydg" Пробег: "497127.000000" Марка: "qgdkn" ФИО: "slizlxsvwrkzzfxtoemnd" Пробег: "499079.000000"</p>
<p>Марка: "whonr" ФИО: "biyyxnkzmbdqcgc" Пробег: "361702.000000" Марка: "nwkutakvw" ФИО: "mmkfptfostysfbxxdfafhnbrjwts" Пробег: "492850.000000" Марка: "geant" ФИО: "sjolkgsifqrksyrhcks" Пробег: "309048.000000" qsort - сортировка</p>	ФИО	2	<p>Марка: "geant" ФИО: "sjolkgsifqrksyrhcks" Пробег: "309048.000000" Марка: "nwkutakvw" ФИО: "mmkfptfostysfbxxdfafhnbrjwts" Пробег: "492850.000000" Марка: "whonr" ФИО: "biyyxnkzmbdqcgc" Пробег: "361702.000000"</p>
<p>Марка: "nstd" ФИО: "zfltbgcprmxmacrmw" Пробег: "107639.000000" Марка: "dtpvsskmorpnnuy" ФИО: "xglmazjmznqsbm" Пробег: "445256.000000" Марка: "p" ФИО: "vmxqykgnxvjxeherveqfsss" Пробег: "486176.000000" Марка: "wuclaaxxsxhy" ФИО: "gvvezekqaojhgbzqlxkiaxic" Пробег: "433740.000000" Марка: "utpu" ФИО: "vrapbpfhfzqmhrnzfljnnix" Пробег: "451295.000000" Марка: "cjxtcpudsjketp"</p>	ФИО	1	<p>Марка: "wuclaaxxsxhy" ФИО: "gvvezekqaojhgbzqlxkiaxic" Пробег: "433740.000000" Марка: "cjxtcpudsjketp" ФИО: "jbndqmibyhoiufnyomstr" Пробег: "428203.000000" Марка: "ubzghsvneyc" ФИО: "oqtsoqizmom" Пробег: "227452.000000" Марка: "p" ФИО: "vmxqykgnxvjxeherveqfsss" Пробег: "486176.000000" Марка: "utpu" ФИО: "vrapbpfhfzqmhrnzfljnnix" Пробег: "451295.000000"</p>

ФИО: "jbndqmibyxoiufnyomstr" Пробег: "428203.000000" Марка: "ubzghsvneyc" ФИО: "oqtsoqizmom" Пробег: "227452.000000" DoubleSelection - сортировка			Марка: "dtpvsskmopnnuy" ФИО: "xglmazjmznqsbm" Пробег: "445256.000000" Марка: "nstd" ФИО: "zfltbgcprmxmacrmw" Пробег: "107639.000000"
Марка: "agarpqnrtamq" ФИО: "sorbfuuzymhxyxlke" Пробег: "420304.000000" Марка: "gdiwdpmgbdv" ФИО: "tpwyxsxvehteeerkskqwuoraedg" Пробег: "228443.000000" Марка: "dbcsxar" ФИО: "anwhidnmuyheqfafwajz" Пробег: "497684.000000" Марка: "imrkgrmyjmn" ФИО: "tvkjehqkxvlcrInt" Пробег: "331720.000000" Марка: "bkvnqozqz" ФИО: "fhharqmyzcizavdtjrmf" Пробег: "419437.000000" DoubleSelection - сортировка	пробег	2	Марка: "dbcsxar" ФИО: "anwhidnmuyheqfafwajz" Пробег: "497684.000000" Марка: "agarpqnrtamq" ФИО: "sorbfuuzymhxyxlke" Пробег: "420304.000000" Марка: "bkvnqozqz" ФИО: "fhharqmyzcizavdtjrmf" Пробег: "419437.000000" Марка: "imrkgrmyjmn" ФИО: "tvkjehqkxvlcrInt" Пробег: "331720.000000" Марка: "gdiwdpmgbdv" ФИО: "tpwyxsxvehteeerkskqwuoraedg" Пробег: "228443.000000"
Марка: "puchjeaisf" ФИО: "bgjushvwruwnaqy" Пробег: "378349.000000" Марка: "dquumybvcbgwhxy" ФИО: "gujppfjmewowwhazz" Пробег: "17804.000000" Марка: "lvwjxapwjowyxqjm" ФИО: "qvtutjsrqtsrpnfkmqkmfgvve" Пробег: "166953.000000" Марка: "wehduezraijrb" ФИО: "kuuxzgjpqx" Пробег: "233976.000000" Марка: "sscncmatte" ФИО: "yfhypsrqxbltbmzkeqicqcwv"	ФИО	2	Вывод массива структур: Марка: "sscncmatte" ФИО: "yfhypsrqxbltbmzkeqicqcwv" Пробег: "354946.000000" Марка: "vnebdcgm" ФИО: "wfuoexbxqyluqvvhzvcre" Пробег: "87741.000000" Марка: "lvwjxapwjowyxqjm" ФИО: "qvtutjsrqtsrpnfkmqkmfgvve" Пробег: "166953.000000" Марка: "wehduezraijrb" ФИО: "kuuxzgjpqx" Пробег: "233976.000000"

<p>Пробег: "354946.000000"</p> <p>Марка: "vnebdcgm"</p> <p>ФИО:</p> <p>"wfuoexbxyqyluqvvhzvcre"</p> <p>Пробег: "87741.000000"</p> <p>Марка: "kbkphxrfenwinxhn"</p> <p>ФИО:</p> <p>"fyjyvifvhmlgtvjekrbeyhtwpgwy"</p> <p>Пробег: "312498.000000"</p> <p>Марка: "fu"</p> <p>ФИО:</p> <p>"dqfilmrzxavujzfccjdlezalvah"</p> <p>Пробег: "22125.000000"</p> <p>bubble - сортировка</p>			<p>Марка: "dquumyvvcgbwhxy"</p> <p>ФИО: "gujppfjmewowwhazz"</p> <p>Пробег: "17804.000000"</p> <p>Марка: "kbkphxrfenwinxhn"</p> <p>ФИО:</p> <p>"fyjyvifvhmlgtvjekrbeyhtwpgwy"</p> <p>Пробег: "312498.000000"</p> <p>Марка: "fu"</p> <p>ФИО:</p> <p>"dqfilmrzxavujzfccjdlezalvah"</p> <p>Пробег: "22125.000000"</p> <p>Марка: "puchjeaisf"</p> <p>ФИО: "bgjushvwruwnaqy"</p> <p>Пробег: "378349.000000"</p>
<p>Марка: "jwxsv"</p> <p>ФИО: "emddcyvvserkq"</p> <p>Пробег: "56383.000000"</p> <p>Марка: "np"</p> <p>ФИО: "gfzltmkfmh"</p> <p>Пробег: "216545.000000"</p> <p>Марка: "gnvlqymo"</p> <p>ФИО: "esoqkbvryueytsuiczu"</p> <p>Пробег: "11567.000000"</p> <p>Марка: "drtbdb"</p> <p>ФИО: "zyvunleqgehaj"</p> <p>Пробег: "175234.000000"</p> <p>Марка: "bcefbaojgfehkp"</p> <p>ФИО: "zqiukxhrqovxoehk"</p> <p>Пробег: "171419.000000"</p> <p>bubble - сортировка</p>	марка	1	<p>Марка: "bcefbaojgfehkp"</p> <p>ФИО: "zqiukxhrqovxoehk"</p> <p>Пробег: "171419.000000"</p> <p>Марка: "drtbdb"</p> <p>ФИО: "zyvunleqgehaj"</p> <p>Пробег: "175234.000000"</p> <p>Марка: "gnvlqymo"</p> <p>ФИО: "esoqkbvryueytsuiczu"</p> <p>Пробег: "11567.000000"</p> <p>Марка: "jwxsv"</p> <p>ФИО: "emddcyvvserkq"</p> <p>Пробег: "56383.000000"</p> <p>Марка: "np"</p> <p>ФИО: "gfzltmkfmh"</p> <p>Пробег: "216545.000000"</p>

Вывод массива структур:
Марка: "e" ФИО: "jcxhugmzyysmvjydksmrzlnllnf" Пробег: "381219.000000"
Марка: "toddssadlofjpxykj" ФИО: "ptqdmprozcbmh" Пробег: "111973.000000"
Марка: "db" ФИО: "gvjlhaqspqqbzorueugwjha" Пробег: "259934.000000"
Марка: "kz" ФИО: "ncivyrgftzxip" Пробег: "232324.000000"
Массив выведен.

Введите команду:

- 1) Ввод данных
 - 2) Вывод данных
 - 3) Генерация массива
 - 4) Сортировки
 - 5) Выход
- 4

- 1) qsort
- 2) bubbleSort
- 3) doubleSelectionSort
- 4) Выход в главное меню

1

Выберите поле:

- 1 - марка
 - 2 - ФИО
 - 3 - пробег
- 1

Выберите направление:

- 1 - по возрастанию
 - 2 - по убыванию
- 2

Введите команду:

- 1) Ввод данных
 - 2) Вывод данных
 - 3) Генерация массива
 - 4) Сортировки
 - 5) Выход
- 2

2

- 1) Вывод на экран
 - 2) Запись в .txt
 - 3) Запись в .bin
 - 4) Выход в главное меню
- 1

Вывод массива структур:

Марка: "toddssadlofjpxykj" ФИО: "ptqdmprozcbmh" Пробег: "111973.000000"
Марка: "kz" ФИО: "ncivyrgftzxip" Пробег: "232324.000000"
Марка: "e" ФИО: "jcxhugmzyysmvjydksmrzlnllnf" Пробег: "381219.000000"
Марка: "db" ФИО: "gvjlhaqspqqbzorueugwjha" Пробег: "259934.000000"
Массив выведен.

На примере показана генерация массива. Так был сгенерирован каждый тест

1) Генерация массива структур заданной длины (1 массив)

2) Исследование сортировки (программа 2)

3) Выход в главное меню

1

Введите количество элементов массива структур: 6

Массив из 6 элементов успешно сгенерирован!

Введите команду:

1) Ввод данных

2) Вывод данных

3) Генерация массива

4) Сортировки

5) Выход

2

1) Вывод на экран

2) Запись в .txt

3) Запись в .bin

4) Выход в главное меню

1

Вывод массива структур:

Марка: "qgdkn" ФИО: "slizlxsvwrkzzfxtoemnd" Пробег: "499079.000000"

Марка: "tjwamgplyaw" ФИО: "ltrjrltqmbphptwex" Пробег: "364349.000000"

Марка: "segusoj" ФИО: "ojbnxwgiorbggewpt" Пробег: "360527.000000"

Марка: "zqijlmrfgfrmtcng" ФИО: "mnjcgnkorienarmsaydoplv" Пробег: "262667.000000"

Марка: "hqgwzikmrosgz" ФИО: "ujnljgzeizhxruuoydg" Пробег: "497127.000000"

Марка: "froricmpdvpn" ФИО: "cyqbczdmyuiuihxo" Пробег: "332750.000000"

Массив выведен.

Массив после сортировки

```
Введите команду:
1) Ввод данных
2) Вывод данных
3) Генерация массива
4) Сортировки
5) Выход
4

1) qsort
2) bubbleSort
3) doubleSelectionSort
4) Выход в главное меню
1

Выберите поле:
1 - марка
2 - ФИО
3 - пробег
3

Выберите направление:
1 - по возрастанию
2 - по убыванию
1

Введите команду:
1) Ввод данных
2) Вывод данных
3) Генерация массива
4) Сортировки
5) Выход
2

1) Вывод на экран
2) Запись в .txt
3) Запись в .bin
4) Выход в главное меню
1
Вывод массива структур:
Марка: "zqijlmlrgfrmtcng" ФИО: "mnjcgnkorienarmsaydoply" Пробег: "262667.000000"
Марка: "froricmpdvpn" ФИО: "cyqbczdmuyiuikhx" Пробег: "332750.000000"
Марка: "segusoj" ФИО: "ojbnxwgiorbggewpt" Пробег: "360527.000000"
Марка: "tjwampgplyaw" ФИО: "ltrjrlltqmbphptwex" Пробег: "364349.000000"
Марка: "hqgwzikmrosgz" ФИО: "ujnljgzeizhxruuooydg" Пробег: "497127.000000"
```

```
Введите команду:
1) Ввод данных
2) Вывод данных
3) Генерация массива
4) Сортировки
5) Выход
3

1) Генерация массива структур заданной длины (1 массив)
2) Исследование сортировки (программа 2)
3) Выход в главное меню
1
Введите количество элементов массива структур: 7
Массив из 7 элементов успешно сгенерирован!

Введите команду:
1) Ввод данных
2) Вывод данных
3) Генерация массива
4) Сортировки
5) Выход
2

1) Вывод на экран
2) Запись в .txt
3) Запись в .bin
4) Выход в главное меню
1
Вывод массива структур:
Марка: "nstd" ФИО: "zfltbgcprmxmaccrmw" Пробег: "107639.000000"
Марка: "dtpvsskmpornnu" ФИО: "xglmazjmznqsbm" Пробег: "445256.000000"
Марка: "p" ФИО: "vmxqykgnxvjxeherveqfsss" Пробег: "486176.000000"
Марка: "wuuclaaxxsxhy" ФИО: "gvvezekqaojhgbzqlxkiaxic" Пробег: "433740.000000"
Марка: "utpri" ФИО: "vpabpflhzqmhrnzfljnnix" Пробег: "451295.000000"
Марка: "cjxtdcupsjketr" ФИО: "jbndqmibyxoifnyomstr" Пробег: "428203.000000"
Марка: "ubzghsvneyc" ФИО: "oqtsoqizmom" Пробег: "227452.000000"
Массив выведен.
```

```
1)qsort
2)bubbleSort
3)doubleSelectionSort
4)Выход в главное меню
3
```

Выберите поле:

```
1 - марка
2 - ФИО
3 - пробег
2
```

Выберите направление:

```
1 - по возрастанию
2 - по убыванию
1
```

Сортировка заняла 0.000000 с

Ведите команду:

```
1)Ввод данных
2)Вывод данных
3)Генерация массива
4)Сортировки
5)Выход
2
```

```
1)Вывод на экран
2)Запись в .txt
3)Запись в .bin
4)Выход в главное меню
1
```

Вывод массива структур:

```
Марка: "wuuelaaxxsxhy" ФИО: "gvvezekqaojhgbzqlxkiaxic" Пробег: "433740.000000"
Марка: "cjxtdcpusjketp" ФИО: "jbndqmibyxoiufnyomstr" Пробег: "428203.000000"
Марка: "ubzghsvneyc" ФИО: "oqtsoqizmom" Пробег: "227452.000000"
Марка: "р" ФИО: "vmxqykgnxvjxeherveqfsss" Пробег: "486176.000000"
Марка: "utri" ФИО: "vpabpfhlhzqmhrnzfljnnix" Пробег: "451295.000000"
Марка: "dtpvsskmorppuy" ФИО: "xglmazjmznqsbm" Пробег: "445256.000000"
Марка: "nstd" ФИО: "zfltbgcprmxmaccrmw" Пробег: "107639.000000"
```

Массив выведен.

Сортировка последнего теста таблицы.

```
Марка: "puchjeaisf" ФИО: "bgjushvwruwnaqy" Пробег: "378349.000000"
Марка: "dquumybvcgbwhxy" ФИО: "gujppfjmewowwhazz" Пробег: "17804.000000"
Марка: "lvwjxarpwjomuhqjm" ФИО: "qvtutjsrqtsrpnfkmqkmfgvve" Пробег: "166953.000000"
Марка: "wehduezraijsrb" ФИО: "kuuxzgjrpqx" Пробег: "233976.000000"
Марка: "sscnmatte" ФИО: "yfhypsrxbltbmzkeqicqcwv" Пробег: "354946.000000"
Марка: "vnebdcm" ФИО: "wfuoexbxuyqyluqvvhzvcre" Пробег: "87741.000000"
Марка: "kbkphxrfenwinxhn" ФИО: "fyjyvifvfhmllgtvjekrbeyhtwpgwy" Пробег: "312498.000000"
Марка: "fu" ФИО: "dqfilmrzxavujzfccjdlezalvah" Пробег: "22125.000000"
Массив выведен.
```

Вывод массива структур:

```
Марка: "sscnmatte" ФИО: "yfhypsrxbltbmzkeqicqcwv" Пробег: "354946.000000"
Марка: "vnebdcm" ФИО: "wfuoexbxuyqyluqvvhzvcre" Пробег: "87741.000000"
Марка: "lvwjxarpwjomuhqjm" ФИО: "qvtutjsrqtsrpnfkmqkmfgvve" Пробег: "166953.000000"
Марка: "wehduezraijsrb" ФИО: "kuuxzgjrpqx" Пробег: "233976.000000"
Марка: "dquumybvcgbwhxy" ФИО: "gujppfjmewowwhazz" Пробег: "17804.000000"
Марка: "kbkphxrfenwinxhn" ФИО: "fyjyvifvfhmllgtvjekrbeyhtwpgwy" Пробег: "312498.000000"
Марка: "fu" ФИО: "dqfilmrzxavujzfccjdlezalvah" Пробег: "22125.000000"
Марка: "puchjeaisf" ФИО: "bgjushvwruwnaqy" Пробег: "378349.000000"
Массив выведен.
```

6. Исследование сортировок.

В ходе лабораторной работы я исследовал предложенные мне сортировки, составил таблицу данных и построил графики. Была написана отдельная программа для замеров времени сортировки случайных массивов.

Вводился параметр m – количество генераций массивов длины n (тоже вводилось пользователем) и высчитывалось среднее время сортировки.

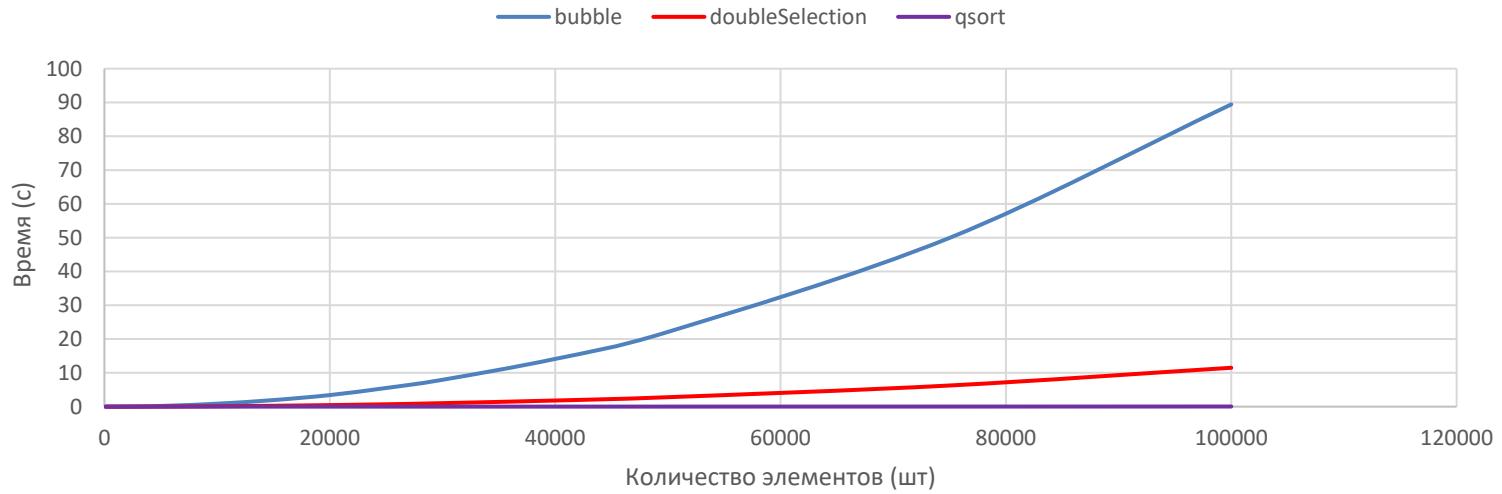
Были получены данные:

m	n	bubble	double	qsort
10000	100	0,000085	0,000019	0,000013
10000	300	0,000706	0,000133	0,000043
5000	700	0,003779	0,000612	0,00011
3000	1000	0,007742	0,001251	0,000162
1000	3000	0,074917	0,010433	0,000551
300	6000	0,309626	0,040807	0,001313
30	10000	0,869998	0,115456	0,002352
20	15000	1,938299	0,257096	0,003479
15	20000	3,423946	0,457485	0,004965
15	25000	5,524411	0,702173	0,00653
15	30000	7,945798	1,029349	0,007744
12	40000	14,12331	1,811529	0,010984
10	50000	22,10765	2,788727	0,014021
5	75000	49,9393	6,255947	0,022959
3	100000	89,43463	11,50737	0,033046

Здесь m – количество генераций массивов, n – количество элементов в массиве. В столбцах с названиями сортировок указано время сортировки в секундах.

По этим данным я построил график:

Зависимость времени сортировки массива структур от количества элементов в нём



Из графика мы видим и это очевидно, что qsort вырвался очень сильно вперед по скорости. doubleSelection довольно сильно оказался быстрее пузырьковой сортировки, ведь его алгоритм по сути есть оптимизированный пузырёк. За счёт уменьшения количества итераций по массиву и одновременного поиска максимума и минимума мы выигрываем по времени.

Ниже приведены примеры того, как я получал данные из таблицы данных выше.

```
[gavrilov.da@unix 5]$ ./test2
Выберите сортировку:
1 - qsort
2 - bubbleSort
3 - doubleSelectionSort
1
Выберите поле:
1 - марка
2 - ФИО
3 - пробег
3
Выберите направление:
1 - по возрастанию
2 - по убыванию
1
Введите количество генераций массива: 5
Введите количество элементов в генерируемом массиве: 75000
Среднее время сортировки одного массива (на входе 5 массивов из 75000 элементов) - 0.023378 с
[gavrilov.da@unix 5]$ █
```

```
[gavrilov.da@unix 5]$ ./test2
Выберите сортировку:
1 - qsort
2 - bubbleSort
3 - doubleSelectionSort
1
Выберите поле:
1 - марка
2 - ФИО
3 - пробег
3
Выберите направление:
1 - по возрастанию
2 - по убыванию
1
Введите количество генераций массива: 10
Введите количество элементов в генерируемом массиве: 50000
Среднее время сортировки одного массива (на входе 10 массивов из 50000 элементов) - 0.014377 с
[gavrilov.da@unix 5]$
```

```
[gavrilov.da@unix 5]$ ./test2
Выберите сортировку:
1 - qsort
2 - bubbleSort
3 - doubleSelectionSort
3
Выберите поле:
1 - марка
2 - ФИО
3 - пробег
3
Выберите направление:
1 - по возрастанию
2 - по убыванию
1
Введите количество генераций массива: 10
Введите количество элементов в генерируемом массиве: 50000
Среднее время сортировки одного массива (на входе 10 массивов из 50000 элементов) - 2.783338 с
[gavrilov.da@unix 5]$
```

```
[gavrilov.da@unix 5]$ ./test2
Выберите сортировку:
1 - qsort
2 - bubbleSort
3 - doubleSelectionSort
2
Выберите поле:
1 - марка
2 - ФИО
3 - пробег
3
Выберите направление:
1 - по возрастанию
2 - по убыванию
1
Введите количество генераций массива: 15
Введите количество элементов в генерируемом массиве: 20000
Среднее время сортировки одного массива (на входе 15 массивов из 20000 элементов) - 3.461254 с
[gavrilov.da@unix 5]$
```

```
[gavrilov.da@unix 5]$ ./test2
Выберите сортировку:
1 - qsort
2 - bubbleSort
3 - doubleSelectionSort
2
Выберите поле:
1 - марка
2 - ФИО
3 - пробег
3
Выберите направление:
1 - по возрастанию
2 - по убыванию
1
Введите количество генераций массива: 3000
Введите количество элементов в генерируемом массиве: 1000
Среднее время сортировки одного массива (на входе 3000 массивов из 1000 элементов) - 0.007755 с
[gavrilov.da@unix 5]$
```

```
[gavrilov.da@unix 5]$ ./test2
Выберите сортировку:
1 - qsort
2 - bubbleSort
3 - doubleSelectionSort
3
Выберите поле:
1 - марка
2 - ФИО
3 - пробег
3
Выберите направление:
1 - по возрастанию
2 - по убыванию
1
Введите количество генераций массива: 10000
Введите количество элементов в генерируемом массиве: 100
Среднее время сортировки одного массива (на входе 10000 массивов из 100 элементов) - 0.000021 с
[gavrilov.da@unix 5]$
```

```
[gavrilov.da@unix 5]$ ./test2
Выберите сортировку:
1 - qsort
2 - bubbleSort
3 - doubleSelectionSort
3
Выберите поле:
1 - марка
2 - ФИО
3 - пробег
3
Выберите направление:
1 - по возрастанию
2 - по убыванию
1
Введите количество генераций массива: 15
Введите количество элементов в генерируемом массиве: 30000
Среднее время сортировки одного массива (на входе 15 массивов из 30000 элементов) - 1.024674 с
[gavrilov.da@unix 5]$
```

```
[gavrilov.da@unix 5]$ ./test2
Выберите сортировку:
1 - qsort
2 - bubbleSort
3 - doubleSelectionSort
1
Выберите поле:
1 - марка
2 - ФИО
3 - пробег
3
Выберите направление:
1 - по возрастанию
2 - по убыванию
1
Введите количество генераций массива: 10000
Введите количество элементов в генерируемом массиве: 100
Среднее время сортировки одного массива (на входе 10000 массивов из 100 элементов) - 0.0000013 с
[gavrilov.da@unix 5]$
```

```
[gavrilov.da@unix 5]$ ./test2
Выберите сортировку:
1 - qsort
2 - bubbleSort
3 - doubleSelectionSort
2
Выберите поле:
1 - марка
2 - ФИО
3 - пробег
3
Выберите направление:
1 - по возрастанию
2 - по убыванию
1
Введите количество генераций массива: 5000
Введите количество элементов в генерируемом массиве: 700
Среднее время сортировки одного массива (на входе 5000 массивов из 700 элементов) - 0.003848 с
[gavrilov.da@unix 5]$
```

```
[gavrilov.da@unix 5]$ ./test2
Выберите сортировку:
1 - qsort
2 - bubbleSort
3 - doubleSelectionSort
1
Выберите поле:
1 - марка
2 - ФИО
3 - пробег
3
Выберите направление:
1 - по возрастанию
2 - по убыванию
1
Введите количество генераций массива: 3
Введите количество элементов в генерируемом массиве: 100000
Среднее время сортировки одного массива (на входе 3 массивов из 100000 элементов) - 0.032046 с
[gavrilov.da@unix 5]$
```

7. Скриншоты работы программы

Команда для сборки программы:

```
cc –o reslab5 lab5.c masfunctions.c sorts.c menu.c
```

Запуск с valgrind:

```
valgrind ./reslab5
```

Также по заданию лабораторной работы были освоены и реализованы запись из и вывод в текстовые и бинарные файлы. Далее будут приложены примеры работы всей программы с записью в файл, считыванием, выводом и генерацией. Всё будет проверено с valgrind.

Генерируем и сортируем 2 массива и записываем их в testTXT.txt и binary.bin.

```
[gavrilov.da@unix 5]$ cc -o reslab5 lab5.c masfunctions.c sorts.c menu.c -g
[gavrilov.da@unix 5]$ valgrind ./reslab5
==17545== Memcheck, a memory error detector
==17545== Copyright (C) 2002-2022, and GNU GPL'd, by Julian Seward et al.
==17545== Using Valgrind-3.22.0 and LibVEX; rerun with -h for copyright info
==17545== Command: ./reslab5
==17545==
```

Ведите команду:

- 1) Ввод данных
 - 2) Вывод данных
 - 3) Генерация массива
 - 4) Сортировки
 - 5) Выход
- 3

1) Генерация массива структур заданной длины (1 массив)

2) Исследование сортировки (программа 2)

3) Выход в главное меню

1
Ведите количество элементов массива структур: 7

Массив из 7 элементов успешно сгенерирован!

Ведите команду:

- 1) Ввод данных
 - 2) Вывод данных
 - 3) Генерация массива
 - 4) Сортировки
 - 5) Выход
- 2

1) Вывод на экран

2) Запись в .txt

3) Запись в .bin

4) Выход в главное меню

1
Вывод массива структур:

Марка: "ktyubvnnp" ФИО: "qdwndffauby" Пробег: "347551.000000"

Марка: "qdqk" ФИО: "losyisslnhaf" Пробег: "226725.000000"

Марка: "i" ФИО: "foqlomooklepe" Пробег: "438568.000000"

Марка: "pggnqzhbopete" ФИО: "ercugnkuby" Пробег: "289824.000000"

Марка: "ewtu" ФИО: "kasarthuiwydavjqxd" Пробег: "76044.000000"

Марка: "nspnzerxzljmo" ФИО: "mhvtbfqcksz" Пробег: "224014.000000"

Марка: "yzkmn" ФИО: "bcegudhhmtvphcm" Пробег: "1805.000000"

Массив выведен.

Введите команду:

- 1) Ввод данных
 - 2) Вывод данных
 - 3) Генерация массива
 - 4) Сортировки
 - 5) Выход
- 4

- 1) quickSort
2) bubbleSort
3) doubleSelectionSort
4) Выход в главное меню
- 2

Выберите поле:

- 1 - марка
 - 2 - ФИО
 - 3 - пробег
- 3

Выберите направление:

- 1 - по возрастанию
- 2 - по убыванию

1

Сортировка заняла 0.002932 с

Введите команду:

- 1) Ввод данных
 - 2) Вывод данных
 - 3) Генерация массива
 - 4) Сортировки
 - 5) Выход
- 2

- 1) Вывод на экран
2) Запись в .txt
3) Запись в .bin
4) Выход в главное меню

1

Вывод массива структур:

Марка: "yzkmn" ФИО: "bcegudhhmtvphcm" Пробег: "1805.000000"
Марка: "ewtu" ФИО: "kasarthuiwydavjqxd" Пробег: "76044.000000"
Марка: "nspnzerxzljmo" ФИО: "mhvtbfqcksz" Пробег: "224014.000000"
Марка: "qdqk" ФИО: "losyisslnhaf" Пробег: "226725.000000"
Марка: "pggnqzhbopete" ФИО: "epcugnkuby" Пробег: "289824.000000"

Вывод массива структур:

Марка: "yzkmn" ФИО: "bcegudhhmtvphcm" Пробег: "1805.000000"
Марка: "ewtu" ФИО: "kasarthuiwydavjqxd" Пробег: "76044.000000"
Марка: "nspnzerxzljmo" ФИО: "mhvtbfqcksz" Пробег: "224014.000000"
Марка: "qdqk" ФИО: "losyisslnhaf" Пробег: "226725.000000"
Марка: "pggnqzhborpete" ФИО: "ercugnkuby" Пробег: "289824.000000"
Марка: "ktyubvnnp" ФИО: "qdwndffauby" Пробег: "347551.000000"
Марка: "i" ФИО: "foqlomooklepe" Пробег: "438568.000000"
Массив выведен.

Введите команду:

- 1) Ввод данных
 - 2) Вывод данных
 - 3) Генерация массива
 - 4) Сортировки
 - 5) Выход
- 2

- 1) Вывод на экран
 - 2) Запись в .txt
 - 3) Запись в .bin
 - 4) Выход в главное меню
- 2

Введите имя текстового файла (+.txt): testTXT.txt

Введите команду:

- 1) Ввод данных
 - 2) Вывод данных
 - 3) Генерация массива
 - 4) Сортировки
 - 5) Выход
- 3

- 1) Генерация массива структур заданной длины (1 массив)
- 2) Исследование сортировки (программа 2)
- 3) Выход в главное меню

1

Введите количество элементов массива структур: 4

Массив из 4 элементов успешно сгенерирован!

Введите команду:

- 1) Ввод данных
- 2) Вывод данных
- 3) Генерация массива
- 4) Сортировки
- 5) Выход

2

|

- 1) Вывод на экран
- 2) Запись в .txt
- 3) Запись в .bin
- 4) Выход в главное меню

1

Вывод массива структур:

Марка: "rtkcllyyj" ФИО: "ivmnwptejwosk" Пробег: "313041.000000"
Марка: "arumuagnmkaxiyi" ФИО: "hfuwelqkwmyoxkezb" Пробег: "157457.000000"
Марка: "vbtkodmnnlwvubrq" ФИО: "cgqyvqmuartdseatykj" Пробег: "387635.000000"
Марка: "xrktmewdwefffwdam" ФИО: "upkpscttytfhxeeeozc" Пробег: "370550.000000"

Массив выведен.

Введите команду:

- 1) Ввод данных
- 2) Вывод данных
- 3) Генерация массива
- 4) Сортировки
- 5) Выход

2

|

- 1) Вывод на экран
- 2) Запись в .txt
- 3) Запись в .bin
- 4) Выход в главное меню

3

Введите имя бинарного файла (+.bin): binary.bin

Введите команду:

- 1) Ввод данных
- 2) Вывод данных
- 3) Генерация массива
- 4) Сортировки
- 5) Выход

4

Введите команду:

- 1) Ввод данных
 - 2) Вывод данных
 - 3) Генерация массива
 - 4) Сортировки
 - 5) Выход
- 4

- 1) quicksort
2) bubbleSort
3) doubleSelectionSort
4) Выход в главное меню
- 3

Выберите поле:

- 1 - марка
 - 2 - ФИО
 - 3 - пробег
- 1

Выберите направление:

- 1 - по возрастанию
 - 2 - по убыванию
- 1

Сортировка заняла 0.005695 с

Введите команду:

- 1) Ввод данных
 - 2) Вывод данных
 - 3) Генерация массива
 - 4) Сортировки
 - 5) Выход
- 2

- 1) Вывод на экран
2) Запись в .txt
3) Запись в .bin
4) Выход в главное меню
- 1

Вывод массива структур:

Марка: "arumuagnmkaxiyi" ФИО: "hfuwelqkwmyoxkezb" Пробег: "157457.000000"

Марка: "rtkcllyyj" ФИО: "ivmnwptejwosk" Пробег: "313041.000000"

Марка: "vbtkodmnnlwvubrq" ФИО: "cgqyvqmuartdseatykj" Пробег: "387635.000000"

Марка: "xrktmewdwefffwdam" ФИО: "upkpscttytfhxeeeoozc" Пробег: "370550.000000"

Массив выведен.

Введите команду:

- 1) Ввод данных
 - 2) Вывод данных
 - 3) Генерация массива
 - 4) Сортировки
 - 5) Выход
- 5

==17545==

==17545== HEAP SUMMARY:

==17545== in use at exit: 0 bytes in 0 blocks

==17545== total heap usage: 31 allocs, 31 frees, 20,146 bytes allocated

==17545==

==17545== All heap blocks were freed -- no leaks are possible

==17545==

==17545== For lists of detected and suppressed errors, rerun with: -s

==17545== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)

[gavrilov.da@unix 5]\$

Файлы testTXT.txt и binary.bin:

```
1 yzkmn
2 bcegudhhmtvphcm
3 1805.000000
4 ewtu
5 kasarthuiwydavjqxd
6 76044.000000
7 nspnzerxzljmo
8 mhvtbfqcksz
9 224014.000000
10 qdqk
11 losyisslnhaf
12 226725.000000
13 pgnqzhbopete
14 epcugnkuby
15 289824.000000
16 ktyubvnnp
17 qdwndffauby
18 347551.000000
19 i
20 foqlomooklepe
21 438568.000000
22
```

```
[gavrilov.da@unix 5]$ hexdump binary.bin
00000000 0009 0000 7472 636b 6c6c 7979 0d6a
00000010 6900 6d76 776e 7470 6a65 6f77 6b73
00000020 4898 000f 0000 7261 6d75 6175 6e67
00000030 7861 7969 1169 0000 6800 7566 6577
00000040 776b 796d 786f 656b 627a c440 4819
00000050 0000 6276 6b74 646f 6e6d 6c6e 7677
00000060 7172 0013 0000 6763 7971 7176 756d
00000070 6474 6573 7461 6b79 606a bd46 1048
00000080 7800 6b72 6d74 7765 7764 6665 7766
00000090 136d 0000 7500 6b70 7370 7463 7974
000000a0 7868 6565 6f6f 637a eec0 48b4
000000ac
[gavrilov.da@unix 5]$
```

8. Трудности при выполнении работы

1. Было обращение к неинициализированным ячейкам памяти, из за того что в структуре перед записью динамической строки (например, при записи массива структур из файла) память не выделялась. Проблема была обнаружена с помощью valgrind и компиляции с флагом -g, валгринд указал строку, где была ошибка.
2. Также при проверки `char *mas` на `NULL` я не учел, что при инициализации стандартно у `mas` есть значение, адрес уже задавался. Поэтому для удобства проверок на наличие инициализированного массива в памяти я обнулял `mas = NULL;` и всё заработало
3. Также при записи структур в бинарный файл приходилось перед записью строки записывать её длину, чтобы потом удачно этот массив считать. Ведь в бинарном файле мы вынуждены как то ограничивать байты информации. В текстовом можно было проходиться и ставить `\n` как разделитель и не помнить длину строки.

9. Выводы

В ходе выполнения данной работы на примере программы, выполняющей сортировку массива структур, были рассмотрены базовые принципы работы со структурами, алгоритмами сортировками и файлами в языке Си:

1. Основные алгоритмы по работе над структурами (объявление, использование, сортировка, запись в файлы) и массивами структур.
2. Организация системы подпрограмм, находящихся в разных файлах.
3. Создание пользовательского интерфейса в виде меню с выбором действий (выбор).
4. Создание и работа с заголовочными файлами (. h) в языке Си.
5. Измерение времени работы программ на Си.
6. Работа с текстовыми и бинарными файлами.
7. Рассмотрены и проанализированы алгоритмы некоторых сортировок.