Министерство науки и образования РФ

Федеральное государственное автономное образовательное учреждение высшего образования «Санкт-Петербургский государственный электротехнический университет «ЛЭТИ»

им. В.И. Ульянова (Ленина)»

Факультет компьютерных технологий и информатики

Кафедра вычислительной техники

**Отчет по лабораторной работе № 1**

**по дисциплине «Программирование» на**

**тему:** **«Структуры в языке C/C++»**

|  |  |
| --- | --- |
| Выполнил студент гр. 9308 | Дементьев Д.П. |
|  |  |
| Проверил к.т.н., доцент | Сискович Т.И |

Санкт-Петербург, 2020

Оглавление

[Введение 3](#_Введение)

[1. Задание 3](#_Toc36047702)

[2. Уточнение задания 3](#_Toc36047703)

[3. Контрольные примеры 5](#_Toc36047704)

[4. Краткое описание алгоритма 6](#_Toc36047705)

[5. Структура вызова функций 7](#_Toc36047706)

[6. Функции 8](#_Toc36047707)

[6.1 Главная функция 8](#_Toc36047708)

[6.2 Функция get\_string 10](#_Toc36047709)

[6.3 Функция get\_int 11](#_Toc36047710)

[6.4 Функция get\_float 12](#_Toc36047711)

[6.5 Функция get\_subject 13](#_Toc36047712)

[6.6 Функция add\_item 14](#_Toc36047713)

[6.7 Функция add\_items 15](#_Toc36047714)

[6.8 Функция print\_tutors 16](#_Toc36047716)

[6.9. Функция selected 17](#_Toc36047717)

[6.10 Функция Menu 1](#_Toc36047718)8

[7. Текст программы 19](#_Toc36047719)

[8. Пример работы программы 26](#_Toc36047720)

8.1.[Исходные данные 26](#_Toc36047721)

8.2.[Вывод программы 27](#_Toc36047722)

[9. Выводы: 29](#_Toc36047723)

# Введение

Целью лабораторной работы является приобретение практических навыков в реализации динамического массива структур на языке программирования C/C++.

# 1. Задание

Написать программу, выполняющую ввод последовательности структур, содержащих характеристики объектов выбранной предметной области, размер которой заранее неизвестен. Осуществить контрольный вывод исходной последовательности, сформировать новую последовательность из элементов исходной, поля которой удовлетворяют заданным условиям и вывести полученный результат.

# 2. Уточнение задания

При выполнении задания необходимо учитывать:

1. Вводимый текст не может быть пустым.
2. Максимальная длина вводимой строки – 24 символа.
3. Пользователь не может обработать, отфильтровать или вывести исходные данные, если он их не ввел.
4. Предметная область - база репетиторов, в которой есть такие критерии как имя, цена за час, предмет преподавания, квалификация, рейтинг.
5. Исходные данные сортируются по критериям предмета, максимальной цены, минимального рейтинга.

Для данной предметной области были выбраны следующие поля структуры:

Таблица 1. Поля структуры tutor

|  |  |  |
| --- | --- | --- |
| **Имя поля** | **Тип** | **Назначение** |
| name | char\* | Имя преподавателя |
| subject | char\* | Предмет преподавания |
| qual | char | Квалификация преподавателя |
| price | int | Цена за час занятия |
| rating | float | Рейтинг преподавателя |

Поле subject определяется введенным пользователем числом от 1 до 5, обозначающим номер предмета:

1. Maths

2. Computer science

3. English

4. Russian

5. Physics

Меню программы должно иметь следующую иерархию:

1 - Ввод исходных данных

2 - Вывод исходных данные

3 - Сортировка исходных данных

4 - Вывод отсортированных данных

5 - Справка

0 - Выход

# 3. Контрольные примеры

Контрольные примеры приведены в таблице 2.

Таблица 2. Контрольные примеры

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № | Поля структуры | | | | | Критерии | |
| **Предмет:** математика**Мин. рейтинг:** 3.0**Макс. цена:** 1000 | **Предмет:** физика**Мин. рейтинг:** 4.0**Макс. цена:** 400 |
| Имя | Предмет | Цена | Рейтинг | Квал. | Вывод | |
| 1 | Alex | Maths | 900 | 5 | A | Исх. структура | - |
| 2 | Vlad | Maths | 350 | 3 | C | Исх. структура | - |
| 3 | Liza | Physics | 500 | 4 | A | - | - |
| 4 |  | English | 780 | 4 | C | Error. You entered empty string. Please, try again. | |
| 5 | Oleg | Russian | abcd | 3 | B | Error reading number. Please, try again. | |

# 4. Краткое описание алгоритма

Начало программы.

Шаг №1. Вывод меню на экран.

Шаг №2. Выбор пользователем одного из пунктов меню.

Шаг №3. Переход к пункту, выбранным пользователем:

1 - Ввод исходных данных. Переход к шагу 4.

2 - Вывод исходных данные. Переход к шагу 4.

3 - Сортировка исходных данных. Переход к шагу 4.

4 - Вывод отсортированных данных. Переход к шагу 4.

5 - Справка. Переход к шагу 4.

0 - Выход. Переход к шагу 4.

Шаг №4. Если пользователь не захотел выйти - переход к шагу 1, иначе конец программы.

Конец.

# 5. Структура вызова функций

Рисунок 1. Структура вызова функций

# дималаба1функции (2)6. Функции

## 6.1 Функция main

**Назначение:**

Является точкой входа в программу.

**Описание переменных:**

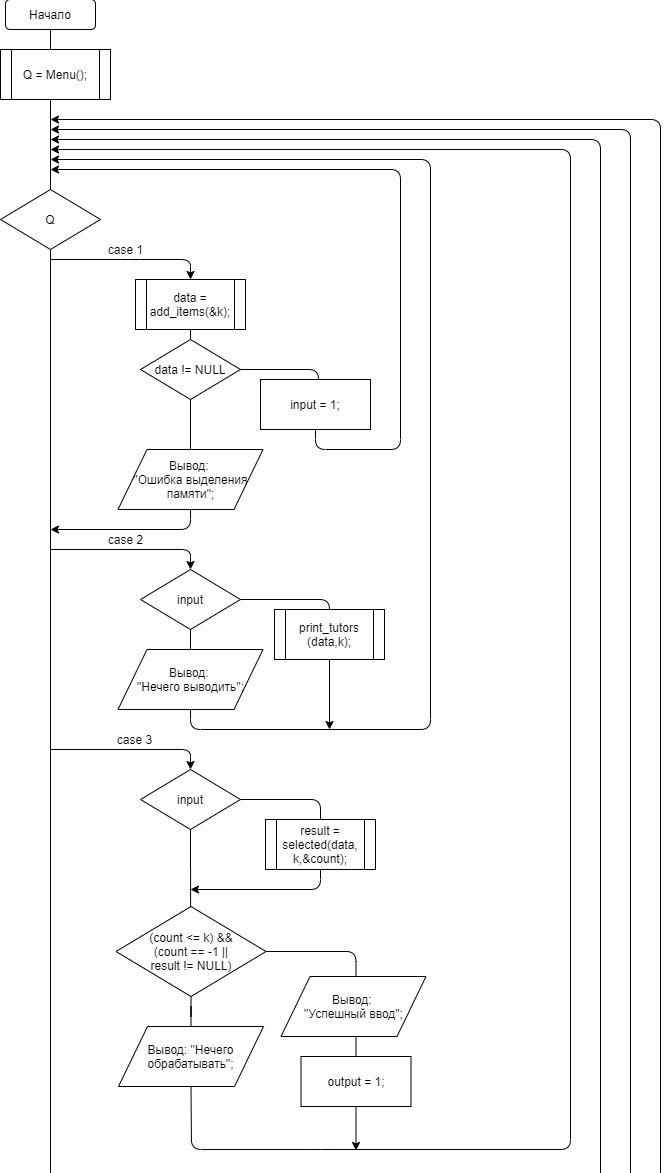
Описание переменных приведено в таблице 3.

Таблица 3. Описание переменных функции main

|  |  |  |
| --- | --- | --- |
| Имя переменной | Тип | Назначение |
| k | int | Количество структур |
| Q | int | Номер пункта меню |
| count | int | Количество записей, подходящих по критериям |
| input | int | Проверка выделения памяти для входных данных |
| output | int | Проверка выделения памяти для выходных данных |
| data | tutor\* | Указатель на структуру, хранящую данные об объекте |
| result | tutor\* | Указатель на новую структуру, хранящую данные об объекте |

**Схема алгоритма:**

Схема алгоритма для функции main представлена на рисунке 2.



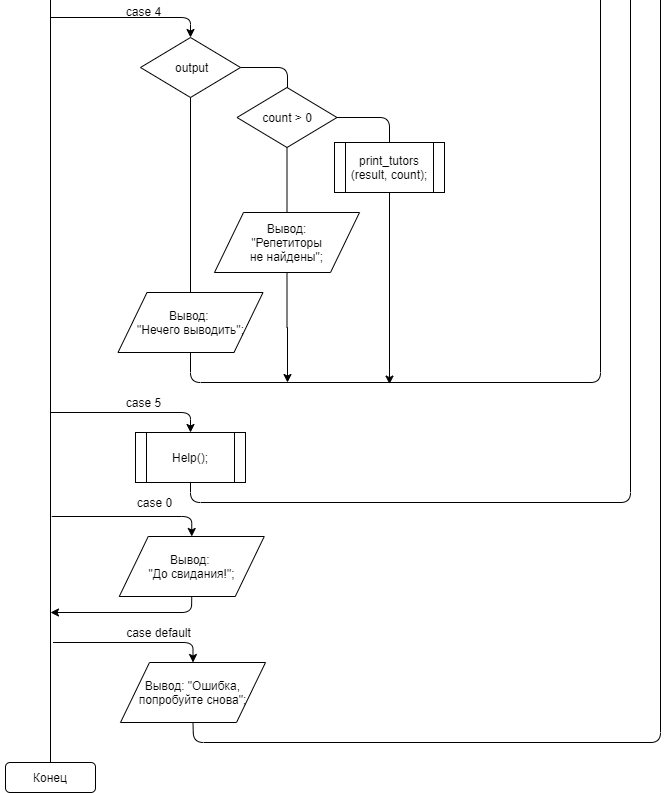


Рис.2. Схема алгоритма

## 6.2 Функция get\_string

**Назначение:**

Функция ввода строки.

**Прототип:**

char \*get\_string();

**Пример вызова:**

list->name = get\_string();

**Описание переменных:**

Описание переменных приведено в таблице 4.

Таблица 4. Описание переменных функции get\_string

|  |  |  |  |
| --- | --- | --- | --- |
| Вид переменной | Имя переменной | Тип | Назначение |
| Локальная | c | char | Символ строки |
| string | сhar\* | Указатель на первый символ строки |
| i | int | Индекс символа строки |

## 6.3 Функция get\_int

**Назначение:**

Функция ввода числа типа int.

**Прототип:**

int get\_int(void);

**Пример вызова:**

list->price = get\_int();

**Описание переменных:**

Описание переменных приведено в таблице 5.

Таблица 5. Описание переменных функции get\_int

|  |  |  |  |
| --- | --- | --- | --- |
| Вид переменной | Имя переменной | Тип | Назначение |
| Локальная | line | char | Буфер |
| curChar | сhar | Переменная для определения корректности ввода |
| temp | int | Проверка успешной записи числа |
| result | int | Сформированное значение типа int |

## 6.4 Функция get\_float

**Назначение:**

Функция ввода числа типа float.

**Прототип:**

int get\_float(void);

**Пример вызова:**

list->rating = get\_float();

**Описание переменных:**

Описание переменных приведено в таблице 6.

Таблица 6. Описание переменных функции get\_float

|  |  |  |  |
| --- | --- | --- | --- |
| Вид переменной | Имя переменной | Тип | Назначение |
| Локальная | line | char | Буфер |
| curChar | char | Переменная для определения корректности ввода |
| temp | int | Проверка успешной записи числа |
| result | float | Сформированное значение типа float |

## 6.5 Функция get\_subject

**Назначение:**

Функция для выбора категории пользователем.

**Прототип:**

сhar \*get\_subject(void);

**Пример вызова:**

List->subject = get\_subject();

**Описание переменных:**

Описание переменных приведено в таблице 7.

Таблица 7. Описание переменных функции get\_subject

|  |  |  |  |
| --- | --- | --- | --- |
| Вид переменной | Имя переменной | Тип | Назначение |
| Локальная | choice | char\* | Указатель на первый символ строки, выбранного предмета |
| c | char | Проверка на конец файла |
| q | int | Номер выбранного предмета |
| i | int | Счетчик цикла |

## 6.6 Функция add\_item

**Назначение:**

Функция добавления новой структуры.

**Прототип:**

void \*add\_item(tutor\*);

**Пример вызова:**

add\_item(data + ++(\*count) -1);

**Описание переменных:**

Описание переменных приведено в таблице 8.

Таблица 8. Описание переменных функции add\_item

|  |  |  |  |
| --- | --- | --- | --- |
| Вид переменной | Имя переменной | Тип | Назначение |
| Формальная | list | tutor\* | Указатель на новую структуру |
| Локальная | с | char | Проверка конца строки |

## 6.7 Функция add\_items

**Назначение:**

Функция добавления структур.

**Прототип:**

tutor \*add\_items(int\*);

**Пример вызова:**

data = add\_items(&k);

**Описание переменных:**

Описание переменных приведено в таблице 9.

Таблица 9. Описание переменных функции add\_items

|  |  |  |  |
| --- | --- | --- | --- |
| Вид переменной | Имя переменной | Тип | Назначение |
| Локальная | data | tutor\* | Указатель на структуру, хранящую данные об объекте |
| exit | char | Проверка окончания добавления структур |
| Формальная | count | int\* | Указатель на количество структур |

## 6.8 Функция print\_tutors

**Назначение:**

Выводит исходные данные.

**Прототип:**

void print\_tutors(tutor\*, int);

**Пример вызова:**

print\_tutors(data, k);

**Описание переменных:**

Описание переменных приведено в таблице 10.

Таблица 10. Описание переменных функции print\_tutors

|  |  |  |  |
| --- | --- | --- | --- |
| Вид переменной | Имя переменной | Тип | Назначение |
| Формальная | list | tutor\* | Указатель на вектор структур |
| count | int | Количество структур |
| Локальная | i | int | Счетчик для цикла |

## 6.9 Функция selected

**Назначение:**

Функция для сортировки структур по заданным критериям.

**Прототип:**

tutor \*selected(tutor\*, int, int\*);

**Пример вызова:**

result = selected(data, k, &count);

**Описание переменных:**

Описание переменных приведено в таблице 11.

Таблица 11. Описание переменных функции selected

|  |  |  |  |
| --- | --- | --- | --- |
| Вид переменной | Имя переменной | Тип | Назначение |
| Локальная | result | tutor\* | Указатель на новую структуру, хранящую данные об объекте |
| i | int | Счетчик цикла |
| maxPrice | int | Критерий суммы |
| minRating | float | Kритерий рейтинга |
| temp | tutor\* | Временная переменная |
| subject | tutor\* | Указатель на критерий предмета |
| Формальная | count | int\* | Указатель на количество структур, удовлетворяющих критериям |
| k | int | Количество структур |
| data | tutor\* | Указатель на структуру, хранящую данные об объекте |

## 6.10 Функция Menu

**Назначение:**

Функция вывода меню.

**Прототип:**

**i**nt Menu();

**Пример вызова:**

Q **=** Menu();

**Описание переменных:**

Описание переменных приведено в таблице 12.

Таблица 12. Описание переменных функции Menu

|  |  |  |  |
| --- | --- | --- | --- |
| Вид переменной | Имя переменной | Тип | Назначение |
| Локальная | Q | int | Номер пункта меню, выбираемый пользователем |

# 8. Текст программы

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <locale.h>

#include <stdlib.h>

#include <malloc.h>

#include <conio.h>

#include <string.h>

#include <windows.h>

#define MAXLEN 24

const char \*subjects[] = {"Maths" , "Computer science", "English", "Russian", "Physics"};

typedef struct

{

// Описание полей

char \*name; // Имя

char \*subject; // Предмет

int price; // Цена за час

char qual; // Квалификация

float rating; // рейтинг преподавателя

} tutor;

char \*get\_string();

int get\_int();

float get\_float();

char \*get\_subject();

void \*add\_item(tutor \*);

tutor \*add\_items(int \*);

void print\_tutors(tutor \*, int);

tutor \*selected(tutor \*, int, int \*);

int Menu();

void Help();

int main()

{

setlocale(LC\_ALL, "RUS");

tutor \*data = NULL, \*result = NULL;

int k, count, Q, input = 0, output = 0;

do {

Q = Menu();

switch (Q)

{

case 1: //input

k = 0;

data = add\_items(&k);

if (data != NULL)

input = 1;

else

printf("Error occured when trying to get memory for array! Please retry.\n");

break;

case 2: //output

if (input)

print\_tutors(data, k);

else

printf("No input to print!\n");

break;

case 3: //process

if (input)

result = selected(data, k, &count);

if ((count <= k) && (count == -1 || result != NULL))

{

printf("Successful selection.\n");

output = 1;

}

else

printf("No input to process!\n");

break;

case 4: //output result

if (output)

if (count > 0) print\_tutors(result, count);

else printf("Tutors not found.\n");

else

printf("No processed data to output!\n");

break;

case 5: //help

Help();

break;

case 0: //exit

printf("Good buy, see you soon!\n");

break;

default:

printf("Error! Try again!\n");

break;

}

system("pause");

} while (Q);

for (Q = 0; Q < k; Q++)

{

free((data + Q)->subject);

free((data + Q)->name);

}

for (Q = 0; Q < count; Q++)

{

free((result + Q)->subject);

free((result + Q)->name);

}

free(data);

free(result);

return 0;

}

char \*get\_string()

{

char c, \*string = (char\*)malloc(MAXLEN\*sizeof(char));

int i=0;

do

{

while ((c=getchar()) != '\n' && i < MAXLEN-1) string[i++] = c;

string[i] = '\0';

if (i < 1) printf("Error. You entered empty string. Please, try again.\n");

} while (i < 1);

if (i >= MAXLEN-1)

while ((c = getchar()) != '\n' && c != EOF);

return string;

}

int get\_int()

{

char line[MAXLEN]; //buffer

char curChar = ' ';

int temp, result;

do {

line[MAXLEN - 1] = '\n';

fgets(line, MAXLEN - 1, stdin);

temp = sscanf(line, "%d%c", &result, &curChar);

temp = !temp || temp < 0 || (curChar != '\n' && curChar != ' ');

if (temp)

printf("Error reading number. Please, try again.\n");

} while (temp); //not a number actually

if (line[MAXLEN - 1] != '\n') //clear all iput

while ((curChar = getchar()) != '\n' && curChar != EOF);

return result;

}

float get\_float()

{

char line[MAXLEN]; //buffer

char curChar = ' ';

int temp;

float result;

do {

line[MAXLEN - 1] = '\n';

fgets(line, MAXLEN - 1, stdin);

temp = sscanf(line, "%f%c", &result, &curChar);

temp = !temp || temp < 0 || (curChar != '\n' && curChar != ' ');

if (temp)

printf("Error reading number. Please, try again.\n");

} while (temp); //not a number actually

if (line[MAXLEN - 1] != '\n') //clear all iput

while ((curChar = getchar()) != '\n' && curChar != EOF);

return result;

}

char \*get\_subject()

{

char \*choice = NULL, c;

int q, i;

for (i=0; i<(int)sizeof(subjects)/sizeof(char\*); i++)

printf("%d. %s\n", i+1, subjects[i]);

do

{

scanf("%d", &q);

if (q <= 0 || q > (int)sizeof(subjects)/sizeof(char\*))

printf("This item doesn't exist. Try again.\n");

} while (q <= 0 || q > (int)sizeof(subjects)/sizeof(char\*));

while ((c = getchar()) != '\n' && c != EOF);

printf("Your chosen %s\n", choice = subjects[q-1]);

return choice;

}

void \*add\_item(tutor \*list)

{

system("cls");

char c;

list->name = (char\*)malloc(MAXLEN\*sizeof(char));

list->subject = (char\*)malloc(MAXLEN\*sizeof(char));

if (list->subject && list->name)

{

printf("Enter tutor's name: \n");

list->name = get\_string();

printf("Enter academic subject: \n");

list->subject = get\_subject();

printf("Enter price per 1 hour(RUB): \n");

list->price = get\_int();

printf("Enter qualifications: \n");

scanf("%c", &(list->qual));

while ((c=getchar()) != '\n' && c != EOF);

do

{

printf("Enter tutor's rating: \n");

list->rating = get\_float();

} while (list->rating <= 0 || list->rating > 5);

}

}

tutor \*add\_items(int \*count)

{

tutor \*data = NULL; // База репетиторов

char exit = '1', c;

\*count = 0;

do

{

data = (tutor\*)realloc(data, (\*count+1)\*sizeof(tutor));

if (data)

{

add\_item(data + ++(\*count) - 1);

printf("Press 'Enter' to stop, press any key to continue.\n");

exit = getch();

}

else exit = 13;

} while (exit != 13);

return data;

}

void print\_tutors(tutor \*list, int count)

{

int i;

printf("| Tutor`s name | Subject |Price per hour(RUB)|Qualification| Rating |\n");

printf("|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_|\n");

for (i = 0; i < count; i++)

{

printf("|%22.22s|%16.16s|%19d|%13c|%8.2f|\n", (list+i)->name, (list+i)->subject, (list+i)->price, (list+i)->qual, (list+i)->rating);

printf("|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_|\n");

}

}

tutor \*selected(tutor \*data, int k, int \*count)

{

tutor \*result = NULL, \*temp = NULL;

int i; // Параметр цикла

int maxPrice = 0;

float minRating = 0;

char \*subject = NULL;

\*count = 0;

printf("Select subject that you need:\n");

subject = get\_subject();

printf("Select max price per hour(RUB): ");

maxPrice = get\_int();

printf("Select min tutor`s rating: ");

minRating = get\_float();

for (i=0; i<k; i++)

{

if (((data+i)->price <= maxPrice) && ((data+i)->rating >= minRating) && (strcmp((data+i)->subject, subject)==0))

{

temp = result;

result = (tutor\*)realloc(result, (\*count+1)\*sizeof(tutor));

if (result)

\*(result + ++(\*count) - 1) = \*(data+i);

else

{

free(temp);

i = \*count = k + 1;

}

}

}

if (\*count==0) \*count = -1;

return result;

}

//Функция меню

//Возвращает номер пункта меню.

int Menu()

{

int Q; // Выбор пользователя

system("cls");

puts("MENU");

puts("1 - Input data");

puts("2 - Output data");

puts("3 - Filter");

puts("4 - Output result");

puts("5 - Help");

puts("0 - Exit");

printf("Select menu item - ");

scanf("%d", &Q);

printf("\n");

fflush(stdin);

return Q;

}

void Help()

{

system("cls");

printf("\tHELP\n");

printf(" We'll help you chosen a tutor. At first, upload(Enter)\ndatabase using menu item 1. Filter database using menu item 3.\n");

printf(" To get the result of program use menu item 4.\n To finish work use menu item 0.\n");

}

# 8. Пример работы программы

## Исходные данные

Таблица 13. Исходные данные

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № | Поля структуры | | | | | Критерии | |
| **Предмет:** математика**Мин. рейтинг:** 3.0**Макс. цена:** 1000 | **Предмет:** физика**Мин. рейтинг:** 4.0**Макс. цена:** 400 |
| Имя | Предмет | Цена | Рейтинг | Квал. | Вывод | |
| 1 | Alex | Maths | 900 | 5 | A | Исх. структура | - |
| 2 | Vlad | Maths | 350 | 3 | C | Исх. структура | - |
| 3 | Liza | Physics | 500 | 4 | A | - | - |
| 4 |  | English | 780 | 4 | C | Error. You entered empty string. Please, try again. | |
| 5 | Oleg | Russian | abcd | 3 | B | Error reading number. Please, try again. | |

## 8.1. Вывод программы

Рисунок 3. Пример работы программы

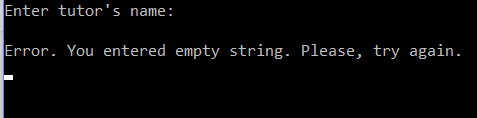
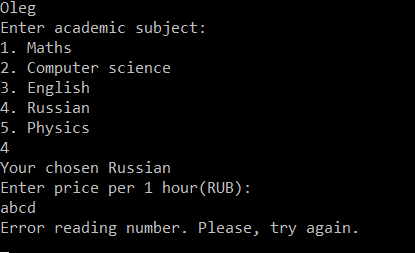


Рисунок 4. Пример работы программы



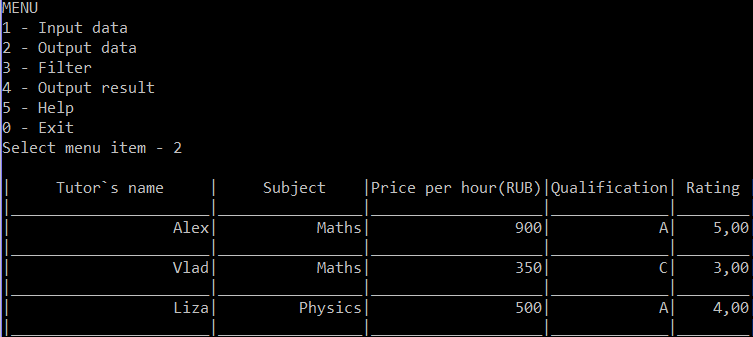
Рисунок 5. Пример работы программы

Рисунок 6. Пример работы программы

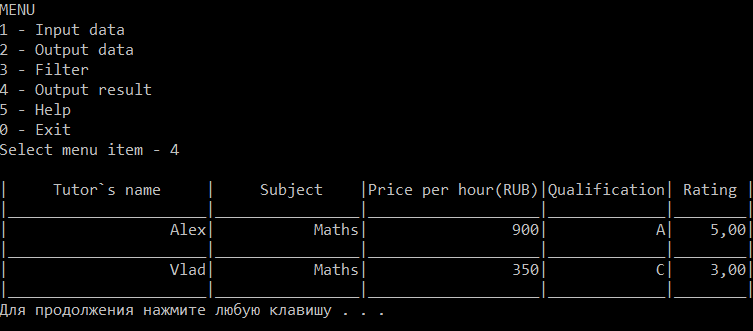
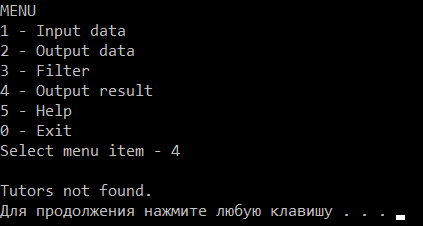


Рисунок 7. Пример работы программы



## 8.2 Результат работы программы

При выполнении программы получены результаты, совпадающие со значениями, приведенными в Таблице 2. Ошибки не обнаружены. Пример протокола выполнения программы приведены на рисунках 3-7.

# 9. Заключение

При выполнении лабораторной работы получены практические навыки в реализации динамического массива структур на языке программирования C/C++.