Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Алтайский государственный технический университет им. И. И. Ползунова»

Факультет информационных технологий
Кафедра прикладной математики

Отчет защищен с оценкой _____

Преподаватель _____
(подпись)
«___» _____ 2026 г.

Отчет
По лабораторной работе №2

«Текст. Массив строк»

по дисциплине «Программирование – 2 семестр»

Студент группы ПИ-54 Дворников М. С.

Преподаватель доцент,к.т.н. Егорова Е.В.

Барнаул 2025

Задание. (№1). Массив строк «Аэропорт»
Каждая строка содержит:
- номер рейса;
- тип самолета;
- пункт назначения;
- дни отправления;
- время вылета;
- время прилета;
- цена билета.

Например: 121 Ту-154 Москва 1,3,5 10.00 14.00 5000.20
Запрос: определить все рейсы до заданного города, отправляющиеся в определенный день на определенном типе самолета.

Текст программы:

```
#define MAX_LEN 255  // Максимальная длина строчки пользователя
#define MAX_NUM 15   // Максимальное количество записей в таблицу

#define COLUMNS_NUM 7  // Количество столбцов
#define COLUMNS_SIZE 11  // Ширина строки в символах


void make_table(const char** row, char mode = 'a') {
    // Функция для отрисовки строки таблицы
    // row: массив указателей на слова, которые нужно вывести
    // mode: какие границы таблички использовать:
    // 'h' - (head) сейчас самый верх таблицы;
    // 'e' - (extra) нужно добавить строку, не отделяя от предыдущей;
    // 'a' - (add) нужно добавить отдельную строчку;
    // 'f' - (final) сейчас самый конец таблицы;
    // 's' - (space) нужно сделать отступ сверху или снизу;

    const char* edge = "-+|";
    switch (mode) {
    case 'h': {
        // Верх таблицы
        printf("%c", edge[1]);
        for (int i = 1; i < COLUMNS_NUM * (COLUMNS_SIZE + 1); i++)
            if (i % (COLUMNS_SIZE + 1)) printf("%c", edge[0]); else printf("%c", edge[1]);
        printf("%c\n", edge[1]);
        // make_table(row, 's');  // Делаем отступ сверху

        // Теперь выводим слова
        printf("%c", edge[2]);
        for (int w = 0; w < COLUMNS_NUM; w++) {
            const char* cur_word = row[w];      // Текущее слово
            int word_len = mystrlen(cur_word);  // Его длина
            if (word_len >= COLUMNS_SIZE)  // Если слово не помещается в столбик
                for (int i = 0; i < COLUMNS_SIZE; i++)
                    printf("%c", cur_word[i]);  // Выводим его посимвольно
            else {
                // Расстояние слева
                int left_space = (COLUMNS_SIZE - word_len) / 2;
                for (int i = 0; i < left_space; i++) printf(" ");
                // Вывод слова
                printf("%s", cur_word);
                // Расстояние справа
                int right_space = COLUMNS_SIZE - word_len - left_space;
                for (int i = 0; i < right_space; i++) printf(" ");
            }
            printf("%c", edge[2]);
```

```
    }
    printf("\n");
    // make_table(row, 's');  // Делаем отступ снизу
    break;
}
case 'e': {
    // Добавляем слова
    printf("%c", edge[2]);
    for (int w = 0; w < COLUMNS_NUM; w++) {
        const char* cur_word = row[w];      // Текущее слово
        int word_len = mystrlen(cur_word);  // Его длина
        if (word_len >= COLUMNS_SIZE)  // Если слово не помещается в столбик
            for (int i = 0; i < COLUMNS_SIZE; i++)
                printf("%c", cur_word[i]);  // Выводим его посимвольно
        else {
            // Расстояние слева
            int left_space = (COLUMNS_SIZE - word_len) / 2;
            for (int i = 0; i < left_space; i++) printf(" ");
            // Вывод слова
            printf("%s", cur_word);
            // Расстояние справа
            int right_space = COLUMNS_SIZE - word_len - left_space;
            for (int i = 0; i < right_space; i++) printf(" ");
        }
        printf("%c", edge[2]);
    }
    printf("\n");
    // make_table(row, 's');  // Делаем отступ снизу
    break;
}
case 'a': {
    // Верх таблицы
    printf("%c", edge[1]);
    for (int i = 1; i < COLUMNS_NUM * (COLUMNS_SIZE + 1); i++)
        if (i % (COLUMNS_SIZE + 1)) printf("%c", edge[0]); else printf("%c", edge[1]);
    printf("%c\n", edge[1]);
    // make_table(row, 's');  // Делаем отступ сверху

    // Теперь выводим слова
    printf("%c", edge[2]);
    for (int w = 0; w < COLUMNS_NUM; w++) {
        const char* cur_word = row[w];      // Текущее слово
        int word_len = mystrlen(cur_word);  // Его длина
        if (word_len >= COLUMNS_SIZE)  // Если слово не помещается в столбик
            for (int i = 0; i < COLUMNS_SIZE; i++)
                printf("%c", cur_word[i]);  // Выводим его посимвольно
        else {
            // Расстояние слева
            int left_space = (COLUMNS_SIZE - word_len) / 2;
            for (int i = 0; i < left_space; i++) printf(" ");
            // Вывод слова
            printf("%s", cur_word);
            // Расстояние справа
            int right_space = COLUMNS_SIZE - word_len - left_space;
            for (int i = 0; i < right_space; i++) printf(" ");
        }
        printf("%c", edge[2]);
    }
    printf("\n");
    // make_table(row, 's');  // Делаем отступ снизу
    break;
}
case 'f': {
    //make_table(row);  // Отрисовываем слова как обычно
    // Низ таблицы
    printf("%c", edge[1]);
```

```
        for (int i = 1; i < COLUMNS_NUM * (COLUMNS_SIZE + 1); i++)
            if (i % (COLUMNS_SIZE + 1)) printf("%c", edge[0]); else printf("%c", edge[1]);
        printf("%c\n", edge[1]);
        break;
    }
    case 's': {
        printf("%c", edge[2]);
        for (int i = 1; i < COLUMNS_NUM * (COLUMNS_SIZE + 1); i++)
            if (i % (COLUMNS_SIZE + 1)) printf(" "); else printf("%c", edge[2]);
        printf("%c\n", edge[2]);
        break;
    }
    }
}


int main() {
    SetConsoleCP(1251); SetConsoleOutputCP(1251);  // Ввод на русском
    system("color F0");
    const char* sep = " ,.?!-;:()/\\|\"\'\n\0";

    // Шапка таблицы
    const char* head[2][COLUMNS_NUM];
    head[0][0] = "номер";  head[1][0] = "рейса";
    head[0][1] = "тип";    head[1][1] = "самолета";
    head[0][2] = "пункт";  head[1][2] = "назначения";
    head[0][3] = "дни";    head[1][3] = "отправления";
    head[0][4] = "время";  head[1][4] = "вылета";
    head[0][5] = "время";  head[1][5] = "прилета";
    head[0][6] = "цена";   head[1][6] = "билета";

    const char* data[MAX_NUM][COLUMNS_NUM];  // Здесь будем хранить всю информацию
    char* user_input;  // Сюда пользователь будет сохранять свой ввод

    printf("Доброе утро! Давайте заполним таблицу.\n");
    printf("Всего можно добавить %d записей. На каждую запись выделяется не более %d
символов\n", MAX_NUM, MAX_LEN);
    printf("Чтобы закончить ввод введите \"###\". Помните, в таблице %d колонок - информацию
нужно разделять пробелом.\n\n", COLUMNS_NUM);
    int longest_word = 0;  // Длина самого длинного слова
    int row;  // Номер ряда
    for (row = 0; row < MAX_NUM;) {
        user_input = (char*)calloc(MAX_LEN + 1, sizeof(char));  // <-?
        if (user_input == NULL) {
            printf("Ошибка: Недостаточно памяти для буфера ввода user_input\n");
            return 1;
        }
        printf("Введите %d запись:\n", row + 1);
        fgets(user_input, MAX_LEN + 1, stdin);
        if (!mystrcmp(user_input, "###\n")) { // Нужно закончить ввод
            if (row) { printf("Ввод закончен\n\n"); free(user_input); break; }
            else { printf("Чтобы принудительно завершить ввод, нужно ввести хотя бы одну
запись\n\n"); free(user_input); continue; }
        }
        if (user_input[mystrlen(user_input) - 1] != '\n') while (getchar() != '\n');  //
Очищаем входной поток
        else user_input[mystrlen(user_input) - 1] = '\0';  // Удаляем '\n' со строчки

        // Теперь разделим строку на слова
        char* cur_row[COLUMNS_NUM];  // Создаём отдельный массив для нашей строчки
        int full_info = 1;  // Флажок "вся нужная информация есть в строке"
        if ((cur_row[0] = strtok(user_input, " ")) == NULL) { printf("Введена пустая строка,
она не идёт в запись\n\n"); free(user_input); continue; }
        for (int r = 1; r < COLUMNS_NUM && full_info; r++) {  // Разделяем всю строку и
проверяем на количество информации
            cur_row[r] = strtok(NULL, " ");
```

```
            if (cur_row[r] == NULL) {
                printf("В строке нехватает информации, она не идёт в запись\n\n");
                full_info = 0;
                free(cur_row[0]);
            }
        }

        if (full_info) {  // Если всё хорошо, то сохраняем строку в наш массив
            for (int i = 0; i < COLUMNS_NUM; i++) {
                data[row][i] = cur_row[i];
                if (mystrlen(cur_row[i]) > longest_word) longest_word = mystrlen(cur_row[i]);
            }
            row += 1;
            printf("Отлично, запись добавлена\n\n");
        }
    }

    printf("Нажмите любую кнопку, чтобы продолжить:\n");
    _getch();

    do {
        system("cls");
        printf("Текущие рейсы:\n");
        make_table(head[0], 'h');
        make_table(head[1], 'e');
        for (int i = 0; i < row; i++) {
            make_table(data[i]);
        }
        make_table(data[row - 1], 'f');

        printf("Доступен запрос:\n");
        printf("Все рейсы до города N, отправляющиеся в день D на типе самолёта P\n");
        char* N, * P;  // Нужные слова
        // "До города N"
        N = (char*)calloc(longest_word + 3, sizeof(char));
        printf("\n\nВведите город N (не более %d символов).\nЧтобы закончить ввод нажмите
enter:\n", longest_word + 2);
        fgets(N, longest_word + 3, stdin);
        if (N[mystrlen(N) - 1] != '\n') while (getchar() != '\n');  // Очищаем входной поток
        else N[mystrlen(N) - 1] = '\0';  // Удаляем '\n' со строчки

        // "В день D"
        char D;
        printf("\n\nВведите день D (1 символ).\nЧтобы закончить ввод нажмите enter:\n");
        scanf("%c", &D);
        while (getchar() != '\n');  // Очищаем входной поток

        // "На типе самолёта P"
        P = (char*)calloc(longest_word + 3, sizeof(char));
        printf("\n\nВведите тип самолёта P (не более %d символов).\nЧтобы закончить ввод
нажмите enter:\n", longest_word + 2);
        fgets(P, longest_word + 3, stdin);
        if (P[mystrlen(P) - 1] != '\n') while (getchar() != '\n');  // Очищаем входной поток
        else P[mystrlen(P) - 1] = '\0';  // Удаляем '\n' со строчки

        int any_trip = 0;  // Нашёлся хотя бы один рейс
        for (int i = 0; i < row; i++) {
            const char* city = data[i][2];  // пункт назначения
            const char* day = data[i][3];  // дни отправления
            const char* plane = data[i][1];  // тип самолёта

            int correct_city = !mystrcmp(city, N);  // Совпадает город
            int correct_day = 0;  // Совпадает день
            for (int j = 0; day[j]; j++)
                if (day[j] != ',' && day[j] == D) correct_day = 1;
            int correct_plane = !mystrcmp(plane, P);  // Совпадает тип самолёта
```

```
            if (correct_city && correct_day && correct_plane) {
                if (!any_trip) {
                    any_trip = 1;
                    printf("\n\nПо вашему запросу получилась следующая таблица:\n");
                    make_table(head[0], 'h');
                    make_table(head[1], 'e');
                    make_table(data[i], 'a');
                }
                else make_table(data[i], 'a');
            }
        }
        if (any_trip) make_table(data[0], 'f');
        else printf("\n\nПо вашему запросу не получилось составить таблицу");

        printf("\n\nНажмите ESC, что бы закончить.\nНажмите любую другую кнопку, чтобы сделать
ещё один запрос:\n");
        //free(text_string);
    } while (_getch() != 27);
    return 0;
}
```

Тесты программы:
Давайте введём в программу следующее расписание:
121 Ту-154 Москва 1,3,5 10.00 14.00 5000.20
122 МиГ Москва 1,5 10.00 14.00 15000.20
123 МиГ Москва 3 14.00 14.00 25000.20
124 МиГ Москва 5 18.00 14.00 35000.20

```
Текущие рейсы:
+-----------+-----------+-----------+-----------+-----------+-----------+-----------+
|   номер   |    тип    |   пункт   |    дни    |   время   |   время   |   цена    |
|   рейса   | самолета  |назначения |отправления|  вылета   |  прилета  |  билета   |
+-----------+-----------+-----------+-----------+-----------+-----------+-----------+
|    121    |  Ту-154   |  Москва   |   1,3,5   |   10.00   |   14.00   |  5000.20  |
+-----------+-----------+-----------+-----------+-----------+-----------+-----------+
|    122    |    МиГ    |  Москва   |    1,5    |   10.00   |   14.00   | 15000.20  |
+-----------+-----------+-----------+-----------+-----------+-----------+-----------+
|    123    |    МиГ    |  Москва   |     3     |   14.00   |   14.00   | 25000.20  |
+-----------+-----------+-----------+-----------+-----------+-----------+-----------+
|    124    |    МиГ    |  Москва   |     5     |   18.00   |   14.00   | 35000.20  |
+-----------+-----------+-----------+-----------+-----------+-----------+-----------+
Доступен запрос:
Все рейсы до города N, отправляющиеся в день D на типе самолёта P
```

Проверим случай, когда по запросу найдена одна строка:

```
Введите город N (не более 10 символов).
Чтобы закончить ввод нажмите enter:
Москва


Введите день D (1 символ).
Чтобы закончить ввод нажмите enter:
3


Введите тип самолёта P (не более 10 символов).
Чтобы закончить ввод нажмите enter:
Ту-154
```

Получаем следующий результат:

```
По вашему запросу получилась следующая таблица:
+-----------+-----------+-----------+-----------+-----------+-----------+-----------+
|   номер   |    тип    |   пункт   |    дни    |   время   |   время   |   цена    |
|   рейса   | самолета  |назначения |отправления|  вылета   |  прилета  |  билета   |
+-----------+-----------+-----------+-----------+-----------+-----------+-----------+
|    121    |  Ту-154   |  Москва   |   1,3,5   |   10.00   |   14.00   |  5000.20  |
+-----------+-----------+-----------+-----------+-----------+-----------+-----------+
```

Проверим случай, когда по запросу найдено несколько строк:

```
Введите город N (не более 10 символов).
Чтобы закончить ввод нажмите enter:
Москва


Введите день D (1 символ).
Чтобы закончить ввод нажмите enter:
5


Введите тип самолёта P (не более 10 символов).
Чтобы закончить ввод нажмите enter:
МиГ


По вашему запросу получилась следующая таблица:
+-----------+-----------+-----------+-----------+-----------+-----------+-----------+
|   номер   |    тип    |   пункт   |    дни    |   время   |   время   |   цена    |
|   рейса   | самолета  |назначения |отправления|  вылета   |  прилета  |  билета   |
+-----------+-----------+-----------+-----------+-----------+-----------+-----------+
|    122    |    МиГ    |  Москва   |    1,5    |   10.00   |   14.00   | 15000.20  |
+-----------+-----------+-----------+-----------+-----------+-----------+-----------+
|    124    |    МиГ    |  Москва   |     5     |   18.00   |   14.00   | 35000.20  |
+-----------+-----------+-----------+-----------+-----------+-----------+-----------+
```

Проверим случай, когда по запросу ничего не найдено:

```
Введите город N (не более 10 символов).
Чтобы закончить ввод нажмите enter:
Москва


Введите день D (1 символ).
Чтобы закончить ввод нажмите enter:
4


Введите тип самолёта P (не более 10 символов).
Чтобы закончить ввод нажмите enter:
МиГ


По вашему запросу не получилось составить таблицу
```