

**Министерство науки и высшего образования Российской Федерации**  
федеральное государственное автономное образовательное учреждение высшего  
образования  
**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»**


**Отчет**

по лабораторной работе No 3 «Лабораторная работа No 3

Строки и ввод-вывод»

Вариант 5-2-3-1.

Автор: Киселёва Анна Николаевна

подпись 

Факультет: ФБИТ

Группа: N3147

Преподаватели:

Грозов Владимир Андреевич



Санкт-Петербург 2025

Задание:

5-

Адрес электронной почты Вида имя@домен.

Примеры:

pupkin@itmo.ru, [rocket+groot@avengers.com](mailto:rocket+groot@avengers.com)

2-

-f=M

Опция -f задает номер объекта данных, начиная с которого нужно осуществлять заданное вариантом преобразование. Объекты данных нумеруются с единицы. Если опция -f не указана, то преобразование выполняется, начиная с первого найденного объекта.

-t=N

Опция -t задает номер последнего объекта данных, над которым нужно осуществлять заданное вариантом преобразование. Если опция -t не указана, преобразование выполняется до последнего найденного объекта данных.

При указании опции -n нумеруются только те объекты данных, которые отвечают критериям поиска, то есть располагаются на одной строке.

3-

Код `Пример`

1

Красный (red) 31

### Содержимое Make-file

.PHONY: all clean

APP=prg3ankN3147

CFLAGS=-Wall -Wextra -Werror -g

all: \$(APP)

\$(APP): \$(APP).c

gcc -o \$(APP) \$(CFLAGS) \$(APP).c

clean:

rm \$(APP)

## Примеры работы программы на различных входных данных:

Ввод не из стандартного потока вывода

```
anna@Huaweianna:~/projects/prg3ankN3147$ ./prg3ankN3147
kiselevanna27@mail.ru lalala
pupupu exampleaddress@gmail.com new
email@address.ru
'kiselevanna27@mail.ru' lalala
pupupu 'exampleaddress@gmail.com' 'new
email@address.ru
'anna@Huaweianna:~/projects/prg3ankN3147$ ./prg3ankN3147 in.txt
```

Ввод из файла

```
anna@Huaweianna:~/projects/prg3ankN3147$ ./prg3ankN3147 in.txt
'kiselevanna27@mail.ru' lalala
pupupu 'exampleaddress@gmail.com' 'new
email@address.ru'anna@Huaweianna:~/projects/prg3ankN3147$ |
```

-v

```
anna@Huaweianna:~/projects/prg3ankN3147$ ./prg3ankN3147 -v
Киселёва Анна Николаевна, гр. N3147
Вариант 5-1-2-3
```

-n

```
anna@Huaweianna:~/projects/prg3ankN3147$ ./prg3ankN3147 -n
kiselevanna27@mail.ru lalala
pupupu exampleaddress@gmail.com new
email@address.ru
'kiselevanna27@mail.ru' lalala
pupupu 'exampleaddress@gmail.com' new
email@address.ru
```

-c -f=M -t=N со стандартного потока вывода

```
anna@Huaweianna:~/projects/prg3ankN3147$ ./prg3ankN3147 -c -f=1 -t=2
kiselevanna27@mail.ru lalala
pupupu exampleaddress@gmail.com new
email@address.ru
kiselevanna27@mail.ru lalala
pupupu exampleaddress@gmail.com new
email@address.ru
anna@Huaweianna:~/projects/prg3ankN3147$ |
```

-c -f=M -t=N из файла

```
anna@Huaweianna:~/projects/prg3ankN3147$ ./prg3ankN3147 in.txt -c -f=1 -t=2
kiselevanna27@mail.ru lalala
pupupu exampleaddress@gmail.com new
email@address.ruanna@Huaweianna:~/projects/prg3ankN3147$ |
```

## Обработка ошибок

```
anna@Huaweianna:~/projects/prg3ankN3147$ ./prg3ankN3147 -f=notnumber
Ошибка: notnumber не является числом.
```

```
anna@Huaweianna:~/projects/prg3ankN3147$ ./prg3ankN3147 -t=notnumber
Ошибка: notnumber не является числом.
```

```
anna@Huaweianna:~/projects/prg3ankN3147$ ./prg3ankN3147 lala.txt
Ошибка: неподдерживаемая опция lala.txt.
anna@Huaweianna:~/projects/prg3ankN3147$ ./prg3ankN3147 -gg
Ошибка: неподдерживаемая опция -gg.
anna@Huaweianna:~/projects/prg3ankN3147$ |
```

## Исходный код

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <regex.h>
#include <unistd.h>

#define BUFSIZE 256

regex_t regex;
int email_count = 0;

int is_valid_email(char *email) { // проверка слова на соответствие email адресу с
помощью масок
    regex_t regex;
    int reti;
    char *pattern = "^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\\.[a-zA-Z]{2,}$"; // маска
    reti = regcomp(&regex, pattern, REG_EXTENDED);
    if (reti) {
        perror("error");
        return EXIT_FAILURE;
    }
    reti = regexec(&regex, email, 0, NULL, 0);
    regfree(&regex);

    return reti == 0; // если слово email то reti == 0
}

void delete_n(char *buf) { // удаление символа \n чтобы корректно обработать
example\n@test.ru
    char *a = buf;
    char *b = buf;

    while (*a) {
        if (*a == '\n' || *a == '\r') { //по другому не придумала как сделать, чтобы была
корректная работа с файлом
```

```

    } else {
        *b++ = *a;
    }
    a++;
}

*b = '\0';
}

void good(char *buffer, int n, int c, int f, int t) {
    char buf[BUFSIZE];
    strcpy(buf, buffer); // копия исходной строки
    if(n == 0) // если введено -n то удалять \n не надо
        delete_n(buf);

    if (is_valid_email(buf)) {
        email_count += 1;
        if (email_count >= f && email_count <= t) { // вывод с условиями флагов -f=M -t=N
            if (c == 1) {
                printf("\e[31m%s\e[0m", buffer); // вывод красным цветом
            }
            else {
                printf("%s", buffer); // вывод обычным выделением
            }
        }
        else {
            printf("%s", buffer); // если слово не email адрес
        }
    }
    else {
        printf("%s", buffer);
    }
}

int is_number(char *a){
    for(int i = 0; a[i] != '\0'; i++){
        if(a[i] > '9' || a[i] < '0')
            return 0;
    }
    return 1;
}

int main(int argc, char *argv[]) {
    int f = 0, t = 100, c = 0, n = 0, v = 0; // считывание флагов
    char *file_name = NULL;
    for (int i = 1; i < argc; i++) {
        if (strcmp(argv[i], "-v") == 0) v = 1;
        if (strcmp(argv[i], "-c") == 0) c = 1;
        if (strcmp(argv[i], "-n") == 0) n = 1;
        if (strncmp(argv[i], "-f=", 3) == 0) {

```

```

    if(is_number(argv[i] + 3) == 0){
        fprintf(stderr, "Ошибка: %s не является числом.\n", argv[i] + 3);
        exit(1);
    }
    f = strtol(argv[i] + 3, NULL, 10);
}
if (strncmp(argv[i], "-t=", 3) == 0) {
    if(is_number(argv[i] + 3) == 0){
        fprintf(stderr, "Ошибка: %s не является числом.\n", argv[i] + 3);
        exit(1);
    }
    t = strtol(argv[i] + 3, NULL, 10);
}

if (i == 1 && access(argv[i], F_OK) != -1) { // Если первый аргумент это файл
    file_name = argv[i];
}
else{
    fprintf(stderr, "Ошибка: неподдерживаемая опция %s.\n", argv[i]);
    exit(1);
}
}

if (v == 1) { // обработка флага -v
    printf("Киселёва Анна Николаевна, гр. N3147\n");
    printf("Вариант 5-1-2-3\n");
    return 0;
}

if (file_name) { //если введено имя файла, то считывание данных из файла
    FILE *file = fopen(argv[1], "r");
    if (file == NULL) {
        fprintf(stderr, "Ошибка: не удалось открыть файл");
        exit(2);
    }
    size_t size = 0;
    size_t s = 10;
    char ch;
    char *buffer = (char *)malloc(s * sizeof(char)); // выделение памяти для вводимых
данных

    while ((ch = fgetc(file)) != EOF) { //ввод из файла
        if (size >= s) {
            s *= 2;
            buffer = (char *)realloc(buffer, s * sizeof(char));
            if (buffer == NULL) {
                perror("error");
                return EXIT_FAILURE;
            }
        }
    }
}

```

```

    }
    buffer[size++] = ch;
}
buffer[size] = '\0'; //конец строки

int flag = 1; // чтобы вывести пробел после каждого слова кроме последнего
char *p = strtok(buffer, " "); // разделение строки по пробелам
while(p){
    if(!flag){
        printf(" ");
    }
    good(p, n, c, f, t); // функция вывода слов
    flag = 0;
    p = strtok(NULL, " "); // переход к новому слову
}

free(buffer); // освобождение памяти
fclose(file); // закрытие файла
return 0;
}
else {
    char ch;
    size_t size = 0;
    size_t s = 10;
    char *buffer = (char *)malloc(s * sizeof(char)); //выделение памяти для вводимых
данных

    if (buffer == NULL) {
        perror("error");
        return EXIT_FAILURE;
    }

    while ((ch = getchar()) != EOF) { // если не из файла
        if (size >= s) {
            s *= 2;
            buffer = (char *)realloc(buffer, s * sizeof(char));
            if (buffer == NULL) {
                perror("error");
                return EXIT_FAILURE;
            }
        }
        buffer[size++] = ch;
    }
    buffer[size] = '\0'; // конец строки
    int flag = 1; // чтобы вывести пробел после каждого слова кроме последнего
    char *p = strtok(buffer, " "); // разделение строки по пробелам
    while(p){
        if(!flag){

```

```
        printf(" ");
    }
    good(p, n, c, f, t); // функция вывода слов
    flag = 0;
    p = strtok(NULL, " "); // переход к новому слову
}
free(buffer);
return 0;
}
}
```