

## Цель работы:

Написать программу для POSIX-совместимой ОС на языке C, выполняющую ввод текстовой информации из файла или стандартного потока ввода, затем осуществляющую преобразование информации в соответствии с вариантом задания и выводящую преобразованную информацию в файл или в стандартный поток вывода.

## Вариант 21.

### Задание:

Найти в исходном тексте все слова, соответствующие заданному шаблону. В шаблоне могут присутствовать символы “.” (точка), означающий любую букву, “@” (коммерческое эт), означающий гласную, “\$” (знак доллара), означающий согласную, и “?” (знак вопроса), означающий любую букву или её отсутствие. Пример: шаблону «.@\$@??» соответствуют слова «mama», «pony», «topic», «aero» и не соответствуют «dad», «port», «aeroplane», «beam».

Шаблон передается программе в командной строке с помощью обязательной опции -совместимой ОС на языке C, выполняющую вводт шаблон. Найденные слова выводятся по одному на строку с указанием номера строки исходного файла, в которой было обнаружено совпадение.

```

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

int main (int argc, char *argv[])
{
    // Для getopt()
    int option=0;                // для получения ключа
    int key_t=0;                 // для проверки, был ли указан ключ -t
    char **input_file = NULL;    // хранит адрес исходного файла, если указан
    char **output_file = NULL;  // хранит адрес конечного файла, если указан
    // Для getline()
    FILE * inFile = NULL;        // ссылка на исходных файл
    size_t len = 0;              // для работы getline
    // Для main()
    int c = 0;                   // для ввода строки
    FILE * outFile = NULL;       // ссылка на конечный файл
    char * mask = NULL;          // шаблон для поиска слова
    char * str = NULL;           // отдельные строки из файла
    char * word = NULL;          // отдельное слово из строки
    int condition = 0;           // для переменного условия главного цикла while
    int yes = 0;                 // кол-во успешно найденных символов при . @ $
    int del = 0;                 // кол-во успешно найденных символов при ?
    int m = 0;                   // счетчик маски
    int mask_count = 0;          // кол-во элементов в маске без ?
    int str_count = 0;           // подсчет номера строки
    char any [] = "abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ"; // любая буква
    char vowel [] = "eaiouyEAIIOUY"; // гласные
    char consonant [] = "bcdfghjklmnpqrstvwxyzBCDFGHJKLMNPQRSTVWXZ"; // согласные

    // Принятие ключей через getopt()
    while ( (option = getopt(argc,argv,"t:h")) != -1)
    {
        switch (option)
        {
            case 't': for (int i=0; i<strlen(optarg); i++)
                        if (strchr(".$%@",optarg[i]) == NULL) { // поиск посторонних символов в введенном шаблоне
                            printf("\nNot allowed symbol(s) in the template\n"
                                    "\nFor more information use option: -h\n\n");
                            return 0; }
                        key_t=1;
                        mask = optarg;
                        break;
            case 'h': printf("h INSERTED\n");
                        break;
            case '?': printf("\nInvalid option: -%c\nUse option: -t \"<template>\" [source_file]\nFor example: -t
                            \".$%@",'\n"
                            "\nFor more information use option: -h\n\n",optopt);
        }
        return 0;
    }
}

// Если нет ключа '-t' для задания шаблона, то выдается ошибка
if (key_t==0) {
    printf("\nOption is required\nUse option: -t \"<template>\" \nFor example: -t \".$%@",'\n"
            "\nFor more information use option: -h\n\n");
    return 0; }

// Считывание кол-ва символов в маске, которые не являются '?'
for (int a=0; a < strlen(mask); a++)
    if ( (mask[a]=='.') || (mask[a]=='@') || (mask[a]=='$') )
        mask_count++;

// Считывание пути исходного файла
input_file = argv + optind;

// Открытие указанного исходного файла для чтения
if (*input_file!=NULL) {
    inFile = fopen(*input_file,"r");
    if (inFile == NULL) {
        printf("Error accessing the IN specified file.\n\n"); // ошибка доступа к файлу
        return 0; } }

// Если исходный файл не был указан, создается и заполняется временный файл
else {
    inFile = tmpfile(); // создание временного файла и получение указателя на файл
    if (inFile==NULL) {
        printf("Unable to create temporary file.");
        return 0; }
    printf("\nInsert your text for scanning:\n");
    c = getchar();
    while (c != EOF) {
        fputc(c, inFile);
        c = getchar(); }
}

```

```

fclose(inFile);
inFile = fopen(inFile, "w+r");           // открытие временного файла
printf("\n\n"); }

// задание условия "проверка на конец файла" для исходного файла
if (inFile!=NULL)
    condition = !feof(inFile);

// Считывание пути конечного файла
output_file = argv + optind + 1;

// Открытие конечного файла для записи и перенаправление в него потока вывода
if ( *output_file!=NULL && strcmp(*output_file,"SHELL=/bin/bash") )
    outFile = freopen(*output_file, "w+", stdout);

while ( condition )
{
    if (inFile!=NULL) {
        getline(&str, &len, inFile); }           // Выделяется строка из исходного файла и записывается в str
    str_count++;                                 // Подсчет строк
    word = strtok(str, " \n\r\t");               // Отделяется первое слово в строке str

    while (word)                                 // Пока не конец строки, читаем слово
    {
        if ( (strlen(word) <= strlen(mask)) && (strlen(word) >= mask_count) ) // Отсекаются заведомо ложные
        {
            yes = del = m = 0;

            // Перебор отделённого слова word[t]
            for (int t=0; t < strlen(mask); t++)
            {
                switch (mask[m])
                {
                    case '.':
                        if (strchr(any,word[t]) && word[t]!='\0') {
                            yes++; m++; }
                        else if (mask[m-1]=='?') {
                            t=t-2; del--; }
                        else
                            yes = del = m = 0;
                        break;
                    case '@':
                        if (strchr(vowel,word[t]) && word[t]!='\0') {
                            yes++; m++; }
                        else if (mask[m-1]=='?') {
                            t=t-2; del--; }
                        else
                            yes = del = m = 0;
                        break;
                    case '$':
                        if (strchr(consonant,word[t]) && word[t]!='\0') {
                            yes++; m++; }
                        else if (mask[m-1]=='?') {
                            t=t-2; del--; }
                        else
                            yes = del = m = 0;
                        break;
                    case '?':
                        if (strchr(any,word[t]) && word[t]!='\0') {
                            del++; m++; }
                        else if (mask[m-1]=='?') {
                            t=t-2; del--; }
                        else if ((word[t]=='\0') && ((yes+del)==strlen(word)) && (yes==mask_count)) {
                            printf("%d\t%s\n", str_count, word);
                            break; }
                        break;
                }

                // Проверка на найденное слово, когда закончилась маска (шаблон)
                if ( ((yes+del)==strlen(word)) && (mask[m]!='.') && (mask[m]!='@') && (mask[m]!='$') &&
                    (mask[m]!='?') ) {
                    printf("%d\t%s\n", str_count, word);
                    break; }
            }
        }

        // Отделяем следующее слово в строке
        word = strtok(NULL, " \n\r\t");
    }

    if (inFile!=NULL)
        condition = !feof(inFile);           // обновление условия "проверка на конец файла" для исходного файла
}

if (inFile!=NULL)
    fclose(inFile);                         // закрытие исходного файла
return 0;
}

```

## Примеры работы программы:

### 1. Случай, если задан исходный файл с текстом.

#### a. Содержимое исходного файла.

```
root@root:~/Documents# cat test.txt
mama pony topic aero dad port aeroplan beam
mamas p0ny TOPiC a ero DAD p0rt AeroPlan Bea2l
```

#### b. Запуск. Через ключ '-t' указывается шаблон для поиска слова. Затем путь к файлу.

```
root@root:~/Documents# ./main -t '.*$@?' test.txt
1      mama
1      pony
1      topic
1      aero
2      mamas
2      TOPiC
root@root:~/Documents# ./main -t '$..?..' test.txt
1      mama
1      pony
1      topic
1      port
1      beam
2      mamas
2      TOPiC
```

### 2. Случай, если задан конечный файл.

#### a. Содержимое исходного файла.

```
root@root:~/Documents# cat test.txt
mama pony topic aero dad port aeroplan beam
mamas p0ny TOPiC a ero DAD p0rt AeroPlan Bea2l
```

#### b. Запуск. Через ключ '-t' указывается шаблон для поиска слова. Затем путь к файлу. Затем путь к конечному файлу.

```
root@root:~/Documents# ./main -t '.*$@?' test.txt out.txt
```

#### c. Содержимое конечного файла.

```
root@root:~/Documents# cat out.txt
1      mama
1      pony
1      topic
1      aero
2      mamas
2      TOPiC
```

### 3. Пример ошибки. Некорректный символ в шаблоне.

```
root@root:~/Documents# ./main -t '.*^$@?' test.txt
```

Not allowed symbol(s) in the template

For more information use option: -h

### 4. Пример ошибки. Некорректный ключ.

```
root@root:~/Documents# ./main -r '.*$@?' test.txt
./main: invalid option -- 'r'
```

Invalid option: -r

Use option: -t '<template>' [source\_file]

For example: -t '.\*\$@?'

For more information use option: -h

### 5. Пример ошибки. Отсутствие необходимого ключа '-t'.

```
root@root:~/Documents# ./main '.*$@?' test.txt
```

Option is required

Use option: -t '<template>'

For example: -t '.\*\$@?'

For more information use option: -h