МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ
РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО
ОБРАЗОВАНИЯ
«Санкт-Петербургский национальный исследовательский
университет информационных технологий, механики и оптики»
Факультет информационных технологий и программирования
Кафедра информационных систем

Программирование
Лабораторная работа № 4

Выполнил студент:
Орлов Александр

Группа: M3107

САНКТ-ПЕТЕРБУРГ
2021

Задача.
Реализовать редактор текстовой метаинформации mp3 файла.
В качестве стандарта метаинформации принимаем ID3v2.
Редактор представлять из себя консольную программу принимающую в качестве аргументов имя файла через параметра --filepath , а также одну из выбранных команд
1. --show - отображение всей метаинформации в виде таблицы
2. --set=prop_name --value=prop_value - выставляет значение определенного поля метаинформации с именем prop_name в значение prop_value
3. --get=prop_name - вывести определенное поле метаинформации с именем prop_name

main.c:

```c
/* Orlov Aleksandr, 12-5, M3107, 22.11.2021 */

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "mp3.h"

int main(int argc, char* argv[])
{
    if (argc < 3 || argc > 4)
    {
        printf("Error: invalid number of arguments");
        return 1;
    }

    struct mp3file* mp3 = calloc(1, sizeof(struct mp3file));
    char arg1[MAX_ARR_SIZE], fileName[MAX_ARR_SIZE];

    sscanf(argv[1], "%11s", arg1);
    if (strcmp(arg1, "--filepath="))
    {
        printf("Error: invalid argument \"%s\"", argv[1]);
        return 1;
    }
    strcpy(fileName, argv[1] + 11);
    if (!mp3open(mp3, fileName))
    {
        printf("Error: unable to open file \"%s\"", fileName);
        return 1;
    }

    char arg2[MAX_ARR_SIZE], propName[MAX_ARR_SIZE];
    char arg3[MAX_ARR_SIZE], propValue[MAX_ARR_SIZE];

    switch (argc)
    {
    case 3:
        sscanf(argv[2], "%6s", arg2);
        if (!strcmp(arg2, "--show"))
        {
            strcpy(propName, argv[2] + 6);
            if (!printFrames(mp3))
```

```
                printf("No information");
        }
        else if (!strcmp(arg2, "--get="))
        {
            strcpy(propName, argv[2] + 6);
            if (!getFrame(mp3, propName))
                printf("No such frame");
        }
        else
        {
            printf("Error: invalid argument \"%s\"", argv[2]);
            return 1;
        }
        break;
    case 4:
        sscanf(argv[2], "%6s", arg2);
        sscanf(argv[3], "%8s", arg3);
        if (strcmp(arg2, "--set="))
        {
            printf("Error: invalid argument \"%s\"", argv[2]);
            return 1;
        }
        if (strcmp(arg3, "--value="))
        {
            printf("Error: invalid argument \"%s\"", argv[3]);
            return 1;
        }
        strcpy(propName, argv[2] + 6);
        strcpy(propValue, argv[3] + 8);
        setFrame(mp3, propName, propValue);
        break;
    }

    mp3close(mp3);

    return 0;
}
```

mp3.c:

```
/* Orlov Aleksandr, 12-5, M3107, 22.11.2021 */

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "mp3.h"

FILE* mp3open(struct mp3file* mp3, char* fileName)
{
    mp3->file = fopen(fileName, "r+");
    if (!mp3->file)
        return mp3->file;

    char c1 = fgetc(mp3->file);
    char c2 = fgetc(mp3->file);
    char c3 = fgetc(mp3->file);

    while (c1 != 'I' || c2 != 'D' || c3 != '3')
    {
        char c1 = fgetc(mp3->file);
        char c2 = fgetc(mp3->file);
        char c3 = fgetc(mp3->file);
        if (c1 == EOF || c2 == EOF || c3 == EOF)
            return 0;
    }
```

```
    mp3->str = malloc(10);
    mp3->str[0] = c1;
    mp3->str[1] = c2;
    mp3->str[2] = c3;

    for (int i = 3; i < 10; ++i)
        mp3->str[i] = fgetc(mp3->file);

    mp3->size = 10 + (mp3->str[6] << 21) + (mp3->str[7] << 14) + (mp3->str[8] << 7) +
mp3->str[9];
    mp3->str = realloc(mp3->str, mp3->size);

    for (int i = 10; i < mp3->size; ++i)
        mp3->str[i] = fgetc(mp3->file);

    return mp3->file;
}

void mp3update(struct mp3file* mp3)
{
    fseek(mp3->file, 0, SEEK_SET);

    for (int i = 0; i < mp3->size; ++i)
        fputc(mp3->str[i], mp3->file);
}

void mp3close(struct mp3file* mp3)
{
    if (mp3 == NULL)
        return;
    fclose(mp3->file);
    free(mp3);
}

int isFrameHeader(char* s)
{
    for (int i = 0; i < 4; ++i)
        if (!(s[i] >= 'A' && s[i] <= 'Z' || s[i] >= '0' && s[i] <= '9'))
            return 0;

    for (int i = 4; i < 8; ++i)
        if (s[i] >= 128)
            return 0;

    if (s[8] & 15 != 0)
        return 0;
    if (s[9] & 128 != 0 || s[9] & 48 != 0)
        return 0;
    return 1;
}

int findFrame(struct mp3file* mp3, char* frameID)
{
    for (int i = 0; i < mp3->size - 10; ++i)
        if (isFrameHeader(mp3->str + i))
        {
            if (mp3->str[i] != frameID[0])
                continue;
            if (mp3->str[i + 1] != frameID[1])
                continue;
            if (mp3->str[i + 2] != frameID[2])
                continue;
            if (mp3->str[i + 3] != frameID[3])
                continue;
```

```
            return i;
        }

    return -1;
}

int printFrames(struct mp3file* mp3)
{
    int count = 0;

    for (int i = 0; i < mp3->size - 10; ++i)
        if (isFrameHeader(mp3->str + i))
        {
            int frameSize = (mp3->str[i + 4] << 21) + (mp3->str[i + 5] << 14) + (mp3-
>str[i + 6] << 7) + mp3->str[i + 7];
            printf("%4s - ", mp3->str + i);
            for (int j = i + 10; j < i + 10 + frameSize; ++j)
                printf("%c", mp3->str[j]);
            printf("\n");
            i += frameSize + 9;
            ++count;
        }

    return count;
}

int getFrame(struct mp3file* mp3, char* frameID)
{
    int pos = findFrame(mp3, frameID);

    if (pos == -1)
        return 0;

    int frameSize = (mp3->str[pos + 4] << 21) + (mp3->str[pos + 5] << 14) + (mp3->str[pos
+ 6] << 7) + mp3->str[pos + 7];

    for (int j = pos + 10; j < pos + 10 + frameSize; ++j)
        printf("%c", mp3->str[j]);

    return 1;
}

void setFrame(struct mp3file* mp3, char* frameID, char* value)
{
    if (!strlen(value))
        return;

    int frameSize;
    int pos = findFrame(mp3, frameID);
    int newFrameSize = 10 + strlen(value) + 1;

    if (pos == -1)
    {
        pos = 10;
        frameSize = 0;
    }
    else
        frameSize = 10 + (mp3->str[pos + 4] << 21) + (mp3->str[pos + 5] << 14) + (mp3-
>str[pos + 6] << 7) + mp3->str[pos + 7];

    char* tmp = malloc(mp3->size - (pos + frameSize));

    memcpy(tmp, mp3->str + pos + frameSize, mp3->size - (pos + frameSize));

    mp3->str[pos] = frameID[0];
```

```
    mp3->str[pos + 1] = frameID[1];
    mp3->str[pos + 2] = frameID[2];
    mp3->str[pos + 3] = frameID[3];

    mp3->str[pos + 4] = ((newFrameSize - 10) >> 21) % 128;
    mp3->str[pos + 5] = ((newFrameSize - 10) >> 14) % 128;
    mp3->str[pos + 6] = ((newFrameSize - 10) >> 7) % 128;
    mp3->str[pos + 7] = (newFrameSize - 10) % 128;

    mp3->str[pos + 8] = 0;
    mp3->str[pos + 9] = 0;

    mp3->str[pos + 10] = 0;

    for (int i = pos + 11; i < pos + newFrameSize; ++i)
        mp3->str[i] = value[i - pos - 11];

    for (int i = pos + newFrameSize; i < mp3->size; ++i)
        mp3->str[i] = tmp[i - pos - newFrameSize];

    free(tmp);
    mp3update(mp3);
}
```

mp3.h:

```
/* Orlov Aleksandr, 12-5, M3107, 22.11.2021 */

#define MAX_ARR_SIZE 1024

struct mp3file
{
    FILE* file;
    char* str;
    int size;
};

FILE* mp3open(struct mp3file*, char*);
void mp3update(struct mp3file*);
void mp3close(struct mp3file*);
int isFrameHeader(char*);
int findFrame(struct mp3file*, char*);
int printFrames(struct mp3file*);
int getFrame(struct mp3file*, char*);
void setFrame(struct mp3file*, char*, char*);
```

Вывод:

В результате выполнения лабораторной работы удалось реализовать программу для редактирования и отображения метаинформации произвольного mp3 файла стандарта ID3V2.