

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

**«САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ
ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»**
Факультет безопасности информационных технологий

Дисциплина:

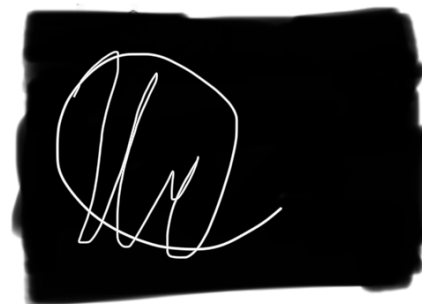
«Программирование»

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №4

Выполнил:

Студент группы N3149

Ильчук Денис.

A black rectangular box containing a white handwritten signature, which appears to be 'Ильчук'.

Проверила:

Горлина А.В.

№ варианта	Тип списка
2	Двусвязный список
№ варианта	Формат данных в строке
8	Дата и время
№ варианта	Команда
2	Команда: filter строка Описание: Удалить из списка элементы, в которых встречается строка.

№ варианта	Формат файла
5	<div><div>Начало файла</div><div><div>cnt</div><div>i₀</div><div>o₀</div><div>...</div><div>i_{N-1}</div><div>o_{N-1}</div><div>data</div><div>p₀</div><div>s₀</div><div>...</div><div>data</div><div>p_{N-1}</div><div>s_{N-1}</div><div>Конец файла</div></div><div><p>Содержимое файла:</p><p>Поле cnt (целое без знака [4 байта]). Количество пар полей индексов и смещений строк, расположенных сразу после этого поля.</p><p>Индексы i₀, ..., i_{N-1} (целые без знака [2 байта]). Индексы соответствующих строк s₀, ..., s_{N-1}, определяющие порядок строк в списке.</p><p>Смещения o₀, ..., o_{N-1} (целые без знака [4 байта]). Смещения полей длин соответствующих строк в файле. Каждый элемент массива o_i содержит смещение в байтах от начала файла поля p_i, за которым следует строка s_i.</p><p>Размеры строк p₀, ..., p_{N-1} (целые без знака [4 байта]). Размеры строк в байтах.</p><p>Строки s₀, ..., s_{N-1}. Строки в кодировке UTF-8 без нулевого символа в конце.</p><p>Размер каждой строки в байтах задается полем p_i.</p><p>Области произвольных данных data. Области произвольного размера (возможно, нулевого).</p></div></div>

Примеры работы программы на различных исходных данных (скриншоты)

```
≡ commands.txt
1  dump
2  filter 23.59.59
3  filter 12.59.39
4  filter 21.07.1931
5  push_front 01.01.01 23.43.10 30.10.2145
6  push_front
7  dump
8  push_back 23.34.45 01.12.1990
9  push_back
10 dump
11 pop_front
12 pop_back
13 dump
14 filter
15 dump
16 dump a.bin a.txt b.bin
17 cat
18 cat a.bin b.bin
19 help
20
```

```
/Users/denisilcuk/Documents/it-17-1/prog/lab4PRG [git::main] [denisilcuk@MacBook-Air-Denis] [15:00]
> ./lab4dviN3149 f.bin < commands.txt
```

```
+-----+
| Program has started |
+-----+
```

```
+-----+
| Ваш список: |
+-----+
```

```
12.59.39
```

```
12.59.21
```

```
01.03.34
```

```
21.05.2024
```

```
22.07.1931
```

```
23.59.59
```

Ваш список:
01.01.01
23.43.10
30.10.2145
12.59.21
01.03.34
21.05.2024
22.07.1931

Ваш список:
01.01.01
23.43.10
30.10.2145
12.59.21
01.03.34
21.05.2024
22.07.1931
23.34.45
01.12.1990

Ваш список:

23.43.10

30.10.2145

12.59.21

01.03.34

21.05.2024

22.07.1931

23.34.45

Ваш список:

23.43.10

30.10.2145

12.59.21

01.03.34

21.05.2024

22.07.1931

23.34.45

Содержимое файла a.bin:

Количество строк = 7

Строка [0] = 23.43.10

Строка [1] = 30.10.2145

Строка [2] = 12.59.21

Строка [3] = 01.03.34

Строка [4] = 21.05.2024

Строка [5] = 22.07.1931

Строка [6] = 23.34.45

Содержимое файла b.bin:

Количество строк = 7

Строка [0] = 23.43.10

Строка [1] = 30.10.2145

Строка [2] = 12.59.21

Строка [3] = 01.03.34

Строка [4] = 21.05.2024

Строка [5] = 22.07.1931

Строка [6] = 23.34.45

Список команд и их описание

`push_front <string 1> <string 2> <string 3> ... <string 100>`

Эта команда добавляет в начало списка строки которые вы ввели.

Количество строк 1 – 100.

Добавляет строки по порядку.

`push_back <string 1> <string 2> <string 3> ... <string 100>`

Эта команда добавляет в конец списка строки которые вы ввели.

Количество строк 1 – 100.

Добавляет строки по порядку.

`dump <filename>`

Записывает в указанный файл элементы из списка, если имя файла пусто, то выводит список на экран.

Можно указать несколько файлов.

`cat <binfilename>`

Выводит на экран элементы, которые сейчас находятся в указанном файле.

Можно указать несколько файлов.

Выводит только файлы в формате .bin

`pop_front`

Удаляет первый элемент в списке.

Если список пуст, то команда ничего не делает.

`pop_back`

Удаляет последний элемент в списке.

Если список пуст, то команда ничего не делает.

`filter <string>`

Удаляет все элементы в которых встречается строка.

Удаляет первый элемент в списке.

Если список пуст, то команда ничего не делает.

`pop_back`

Удаляет последний элемент в списке.

Если список пуст, то команда ничего не делает.

`filter <string>`

Удаляет все элементы в которых встречается строка.

Program has ended successfully

Исходный текст программы с комментариями


```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>
#include <regex.h>
#include "My_Lib.h"
```

```
int main(int argc, char* argv[])
```

```
{
    char *binfilename;
    char command[256];
    Node *head = NULL;
```

```
    if (argc == 2 && strcmp(argv[1], "-v") == 0) //ну тут понятно
```

```
    {
        printf("%s", "\tДенис Ильчук Витальевич, гр. N3149\n\tВариант: 2-8-2-5\n");
        return 0;
    }
```

```
    else if (argc == 2 && (Is_Valid_File_Name(argv[1]) == 0))
```

```
    {
        Start();
        binfilename = argv[1];
    }
```

```
    else
    {
        Errors(1);
    }
```

```
    Read_And_Append_To_List(binfilename, &head); // сразу и список создается и в него из файла читается
```

```
    while (1)//цикл для того чтобы можно было сколько хочешь команд дать
```

```
    {
        if(fgets(command, sizeof(command), stdin) == NULL) // чтобы control+D работал и с EOF тоже подружить
```

```
        {
            End();
```

```
        }
        remove_last_character(command); //удаляет последний \n из команды
        char *token = strtok(command, " "); // считываем программу
```

```
        if (strcmp(token, "cat") == 0)//сам добавил для дебага, но решил оставить
```

```
        {
            while (1) // сколько хочешь файлов можно посмотреть
```

```
            {
                token = strtok(NULL, " ");
                if(token == NULL)
                {
                    break;
                }
            }
```

```
            Print_File(token);
```

```
        }
```

```

}
else if (strcmp(token, "push_front") == 0) // заполняется массив и в
обратном порядке добавляется в список
{
    char *array[100] = {NULL};
    char *arg;
    int i = 0;
    while(1)
    {
        token = strtok(NULL, " ");
        if(token == NULL)
        {
            break;
        }

        arg = token;
        if (Is_Valid_Date_Time_Format(arg) == 0)
        {
            Errors(9);
        }

        Insert_String(array, arg, i);
        i += 1;
    }
    for (int j = i - 1; j >= 0; j--)
    {
        Push_Front(array[j], &head);
    }

    Prepare_List(&head);
}
else if (strcmp(token, "dump") == 0) // я конечно на искосок читал, но
посмотреть список и записать в файл можно
{
    Prepare_List(&head);

    token = strtok(NULL, " ");
    if (token == NULL)
    {
        Print_List(head);
    }
    else
    {
        while(1)
        {
            Txt_Or_Bin(token, &head);
            token = strtok(NULL, " ");
            if(token == NULL)
            {
                break;
            }
        }
    }
}
else if (strcmp(token, "push_back") == 0) // как push_front только
добавляется последовательно
{
    char *arg;

```

```

while(1)
{
    token = strtok(NULL, " ");
    if(token == NULL)
    {
        break;
    }

    arg = token;
    if (Is_Valid_Date_Time_Format(arg) == 0)
    {
        Errors(9);
    }
    Push_Back(arg, &head);
}

Prepare_List(&head);
}
else if (strcmp(token, "pop_front") == 0)
{
    Remove_First(&head);

    Prepare_List(&head);
}
else if (strcmp(token, "pop_back") == 0)
{
    Remove_Last(&head);

    Prepare_List(&head);
}
else if (strcmp(token, "help") == 0)
{
    Print_Help_Message();
}
else if (strcmp(token, "filter") == 0)
{
    token = strtok(NULL, " ");
    if(token == NULL)
    {
    }
    else
    {
        Delete_Node(token, &head);
    }
}
else
{
    printf("\n+-----+
\n\tНеизвестная команда. Список команд доступен при вводе 
help\n+-----+\n");
}
}

Free_List(head);
return 0;
}

```

Файл My_Lib.h:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>
#include <regex.h>
#include "My_Lib.h"
```

```
int main(int argc, char* argv[])
```

```
{
    char *binfilename;
    char command[256];
    Node *head = NULL;
```

```
    if (argc == 2 && strcmp(argv[1], "-v") == 0) //ну тут понятно
    {
```

```
        printf("%s", "\tДенис Ильчук Витальевич, гр. N3149\n\tВариант: 2-8-2-5\n");
        return 0;
    }
```

```
    else if (argc == 2 && (Is_Valid_File_Name(argv[1]) == 0))
    {
```

```
        Start();
        binfilename = argv[1];
    }
```

```
    else
    {
        Errors(1);
    }
```

```
    Read_And_Append_To_List(binfilename, &head); // сразу и список создается и в него из файла читается
```

```
    while (1)//цикл для того чтобы можно было сколько хочешь команд дать
    {
```

```
        if(fgets(command, sizeof(command), stdin) == NULL) // чтобы control+D работал и с EOF тоже подружить
```

```
        {
            End();
        }
```

```
        remove_last_character(command); //удаляет последний \n из команды
        char *token = strtok(command, " "); // считываем программу
```

```
        if (strcmp(token, "cat") == 0)//сам добавил для дебага, но решил оставить
        {
```

```
            while (1) // сколько хочешь файлов можно посмотреть
            {
```

```
                token = strtok(NULL, " ");
                if(token == NULL)
```

```
                {
                    break;
                }
```

```
                Print_File(token);
```

```

    }
}
else if (strcmp(token, "push_front") == 0) // заполняется массив и в
обратном порядке добавляется в список
{
    char *array[100] = {NULL};
    char *arg;
    int i = 0;
    while(1)
    {
        token = strtok(NULL, " ");
        if(token == NULL)
        {
            break;
        }

        arg = token;
        if (Is_Valid_Date_Time_Format(arg) == 0)
        {
            Errors(9);
        }

        Insert_String(array, arg, i);
        i += 1;
    }
    for (int j = i - 1; j >= 0; j--)
    {
        Push_Front(array[j], &head);
    }

    Prepare_List(&head);
}
else if (strcmp(token, "dump") == 0) // я конечно на искосок читал, но
посмотреть список и записать в файл можно
{
    Prepare_List(&head);

    token = strtok(NULL, " ");
    if (token == NULL)
    {
        Print_List(head);
    }
    else
    {
        while(1)
        {
            Txt_Or_Bin(token, &head);
            token = strtok(NULL, " ");
            if(token == NULL)
            {
                break;
            }
        }
    }
}
else if (strcmp(token, "push_back") == 0) // как push_front только
добавляется последовательно
{

```



```

char *arg;
while(1)
{
    token = strtok(NULL, " ");
    if(token == NULL)
    {
        break;
    }

    arg = token;
    if (Is_Valid_Date_Time_Format(arg) == 0)
    {
        Errors(9);
    }
    Push_Back(arg, &head);
}

    Prepare_List(&head);
}
else if (strcmp(token, "pop_front") == 0)
{
    Remove_First(&head);

    Prepare_List(&head);
}
else if (strcmp(token, "pop_back") == 0)
{
    Remove_Last(&head);

    Prepare_List(&head);
}
else if (strcmp(token, "help") == 0)
{
    Print_Help_Message();
}
else if (strcmp(token, "filter") == 0)
{
    token = strtok(NULL, " ");
    if(token == NULL)
    {
    }
    else
    {
        Delete_Node(token, &head);
    }
}
else
{
}

printf("\n+-----+
\n\tНеизвестная команда. Список команд доступен при вводе 
help\n+-----+\n");
}
}

Free_List(head);
return 0;
}

```