

Правительство Российской Федерации
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»
(НИУ ВШЭ)

Московский институт электроники и математики им. А.Н. Тихонова

Практическая работа №1 по дисциплине
«Системное программирование»
Тема работы: «Работа с памятью.
Указатели, массивы, битовые поля, объединения»

Студент гр. БИБ 201
Морин Денис Александрович
«22» ноября 2022 г.

Руководитель
Преподаватель В.И.Морозов
«___» _____ 2022 г.

Москва, 2022

Оглавление

1 Цели и задачи практической работы.....	3
2 Ход работы.....	4
2.1 Программа по зашифрованию данных	4
2.2 Программа расшифрования данных.....	9
3 Вывод.....	11
Приложение 1.....	14
Приложение 2.....	16

1 Цель работы.

Целью данной работы является работа с памятью, указателями, массивами, объединениями.

2 Ход работы

2.1 Программа по зашифрованию данных

Данная программа содержит структуру Item, в которой хранится наименование товара и его размер, представленная на рисунке 1.

```
9 struct Item{
10     int sizeArr[QUANTITY];
11     char name[MAX_NAME_SIZE];
12 };
13
```

Рисунок 1 - Структура Item на языке Си

Для сериализации данных используется объединение U, в котором находятся структура Item с именем item и массив байт arrB, представленные на рисунке 2.

```
15 union U{
16     struct Item item;
17     char arrB[sizeof(struct Item)];
18 };
19
```

Рисунок 2 - Объединение U на языке Си

Согласно моему варианту, мне предстояло зашифровать данные, вычитая из численного представления каждого байта число 10. Эту задачу реализует функция encrypt, которая принимает на вход байт данных и выводит зашифрованный байт. данная функция представлена на рисунке 3.

```
21 char encrypt(char byte){
22     byte -= 10;
23     return byte;
24 }
```

Рисунок 3 - Функция зашифрования байта

Функция `encBuffer`, которая принимает на вход указатели на входной и выходной буферы, функцию обратного вызова и их размер, представлена на рисунке 4.

```
27 void encBuffer(const char * inBuffer, char * outBuffer,  
28               char (*ePtr)(char ), unsigned int len){  
29     for (int i = 0; i < len; i++){  
30         *(outBuffer + i) = ePtr(*(inBuffer + i));  
31     }  
32 }
```

Рисунок 4 - Функция для записи зашифрованных байтов в буфер

С помощью функции `encrypt` она записывает в выходной буфер результаты шифрования над байтами в буфере.

Вышеупомянутые функции, структура и объединение помогают реализовать поставленную задачу, основная идея которой представлена в функции `main` (рисунок 5).

```

35 ▶ int main(int argc, char **argv){
36
37     if (argc != 2){
38         printf( format: "Expected two arguments: the running file "
39                 "and the file to save data");
40         return ERROR_INPUT_DATA;
41     }
42
43     union U un;
44     int maxsize = sizeof(struct Item);
45     for (int i = 0; i < maxsize; i++){
46         un.arrB[i] = 20;
47     }
48     printf( format: "Input item name: ");
49     fgets( Buf: un.item.name, MaxCount: MAX_NAME_SIZE, File: stdin);
50
51     for (int i = 0; i < MAX_NAME_SIZE; i++){
52         if(un.item.name[i] == '\n') {
53             un.item.name[i] = 0;
54             break;
55         }
56     }
57
58     printf( format: "Enter a three integer numbers, where the first is length, "
59             "the second is width and the last is height.\n");

```

Рисунок 5 - Главная функция main

На рисунке 5 показано, что в функцию мы должны передать два аргумента, после чего запустится процесс записи данных. Пользователю требуется ввести наименование товара, а также его длину, ширину и высоту.

Запрос на ввод последних трёх показателей товара представлен на рисунке 6.

```

63     printf( format: "length: ");
64     scanf( format: "%d", &(un.item.sizeArr)[0]);
65
66     printf( format: "width: ");
67     scanf( format: "%d", &(un.item.sizeArr)[1]);
68
69     printf( format: "height: ");
70     scanf( format: "%d", &(un.item.sizeArr)[2]);
71

```

Рисунок 6 - Запрос программы на ввод размеров товара

Далее программа выделяет память для буфера, в который записывает зашифрованные данные. Записывает их в файл, закрывает его и очищает выделенную память (рисунок 7).

```

79     char * buffer = (char *) (malloc( Size: maxsize * sizeof(char)));
80     encBuffer( inBuffer: un.arrB, outBuffer: buffer, ePtr: encrypt, len: maxsize);
81
82     FILE *f = fopen( Filename: argv[1], Mode: "wb");
83     fputs( Str: buffer, File: f);
84
85     free( Memory: buffer);
86     fclose( File: f);
87     return 0;
88 }

```

Рисунок 7 - Часть функции main, реализующая работу с памятью

Программа может обработать несколько ошибок:

- 1) Ошибка, когда передаётся неверное число аргументов, отличное от двух. Например, три, как показано на рисунке 8;

```

PS C:\Users\lkey2\CLionProjects\untitled1\cmake-build-debug> ./untitled1.exe file.bin extraArgument
Expected two arguments: the running file and the file to save data
PS C:\Users\lkey2\CLionProjects\untitled1\cmake-build-debug>

```

Рисунок 8 - Ошибка из-за введения третьего аргумента

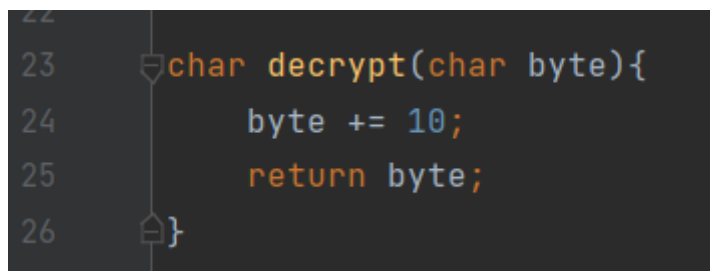
- 2) Ошибка, связанная с передачей хотя бы одного из параметров размера товара, равного 10 (потому что если не обработать эту ошибку, то при шифровании получится символ \0, который будет означать конец строки и данные будут отображаться некорректно). Данная ошибка представлена на рисунке 9.

```
PS C:\Users\lkey2\CLionProjects\untitled1\cmake-build-debug> ./untitled1.exe file.txt
Input item name: Bear
Enter a three integer numbers, where the first is length, the second is width and the last is height.
But it cannot be equal 10.
length: 10
width: 8
height: 5
Input data is consisted a forbidden number (10), so the program ends.
PS C:\Users\lkey2\CLionProjects\untitled1\cmake-build-debug> █
```

Рисунок 9 - Передача запрещенного значения размера

2.2 Программа расшифрования данных

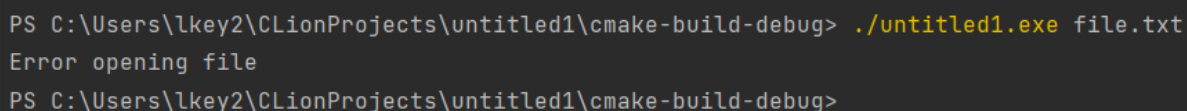
Данная программа схожа с описанной ранее программой зашифрования, а именно такая же структура, такое же объединение, функция записи результата применения функции зашифрования. Небольшое отличие есть только в последней, где мы уже прибавляем к каждому байту 10 для расшифрования (рисунок 10).



```
23 char decrypt(char byte){  
24     byte += 10;  
25     return byte;  
26 }
```

Рисунок 10 - Функция расшифрования

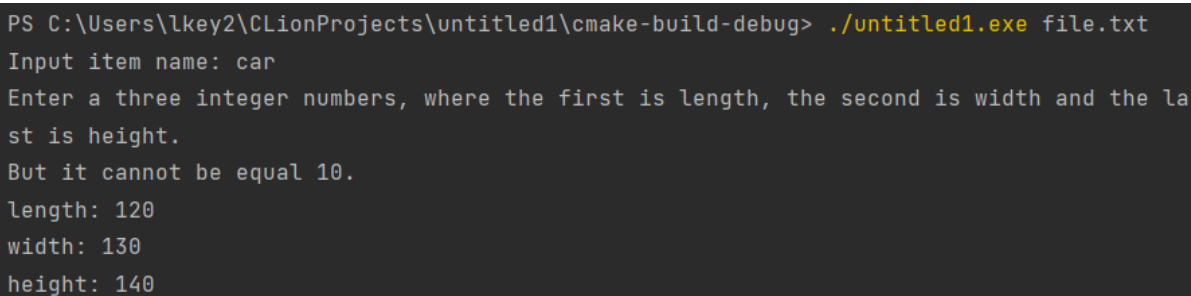
Кроме проиллюстрированный ранее ошибки с неправильным числом аргументов для запуска программы, можно отметить ошибку открытия файла для расшифрования (рисунок 11).



```
PS C:\Users\lkey2\CLionProjects\untitled1\cmake-build-debug> ./untitled1.exe file.txt  
Error opening file  
PS C:\Users\lkey2\CLionProjects\untitled1\cmake-build-debug>
```

Рисунок 11 - ошибка при считывании файла

Наглядно будет продемонстрировать примеры ввода вместе с выводом программы, чтобы убедиться в работоспособности кода. Запустим программу через консоль с помощью команды `./untitled1.exe file.txt` и введем спрашиваемые значения (рисунок 12).



```
PS C:\Users\lkey2\CLionProjects\untitled1\cmake-build-debug> ./untitled1.exe file.txt  
Input item name: car  
Enter a three integer numbers, where the first is length, the second is width and the last is height.  
But it cannot be equal 10.  
length: 120  
width: 130  
height: 140
```

Рисунок 12 - Пример корректного ввода для данной программы

Аналогичным образом запустим программу по расшифровке информации. Результаты представлены на рисунке 13.

```
PS C:\Users\lkey2\CLionProjects\untitled1\cmake-build-debug> ./untitled1.exe file.txt
car
1. 120
2. 130
3. 140
PS C:\Users\lkey2\CLionProjects\untitled1\cmake-build-debug> 
```

Рисунок 13 - Расшифровка данных, введенных недавно

Еще один пример корректной работы программы шифрования, представленный на рисунке 14.

```
PS C:\Users\lkey2\CLionProjects\untitled1\cmake-build-debug> ./untitled1.exe file.txt
Input item name: TEST_MY_SKILL
Enter a three integer numbers, where the first is length, the second is width and the last is height.
But it cannot be equal 10.
length: 1
width: 2
height: 3
PS C:\Users\lkey2\CLionProjects\untitled1\cmake-build-debug> 
```

Рисунок 14 - Результат работы записи в файл file.txt
зашифрованных данных

Расшифрование данных из file.txt, куда были записаны данные выше, представлено на рисунке 15.

```
PS C:\Users\lkey2\CLionProjects\untitled1\cmake-build-debug> ./untitled1.exe file.txt
TEST_MY_SKILL
1. 1
2. 2
3. 3
PS C:\Users\lkey2\CLionProjects\untitled1\cmake-build-debug> 
```

Рисунок 15 - Результат расшифровки из файла

3 Вывод

На этой практике было изучена работа с массивами, указателями, объединениями, а также работа с памятью на языке программирования Си.

Полные коды программ зашифрования и расшифрования можно найти в приложениях 1 и 2 соответственно.

ПРИЛОЖЕНИЕ 1

```
#include <stdio.h>
#include <malloc.h>
#include <stdlib.h>
#define MAX_NAME_SIZE 1000
#define QUANTITY 3
#define ERROR_INPUT_DATA -1

struct Item{
    int sizeArr[QUANTITY];
    char name[MAX_NAME_SIZE];
};

union U{
    struct Item item;
    char arrB[sizeof(struct Item)];
};

char encrypt(char byte){
    byte -= 10;
    return byte;
}
```

```

void encBuffer(const char * inBuffer, char * outBuffer, unsigned int len, char
(*ePtr)(char )){
    for (int i = 0; i < len; i++){
        *(outBuffer + i) = ePtr(*(inBuffer + i));
    }
}

```

```

int main(int argc, char **argv){

    if (argc != 2){
        printf("Expected two arguments: the running file and the file to save
data");
        return ERROR_INPUT_DATA;
    }

```

```

    union U un;
    int maxsize = sizeof(struct Item);
    for (int i = 0; i < maxsize; i++){
        un.arrB[i] = 20;
    }
    printf("Input item name: ");
    fgets(un.item.name, MAX_NAME_SIZE, stdin);

    for (int i = 0; i < MAX_NAME_SIZE; i++){
        if(un.item.name[i] == '\n') {
            un.item.name[i] = 0;
            break;
        }
    }

```

```
}
```

```
printf("Enter a three integer numbers, where the first is length, "  
      "the second is width and the last is height.\n");
```

```
printf("But it cannot be equal 10.\n");
```

```
printf("length: ");  
scanf("%d", &(un.item.sizeArr[0]));
```

```
printf("width: ");  
scanf("%d", &(un.item.sizeArr[1]));
```

```
printf("height: ");  
scanf("%d", &(un.item.sizeArr[2]));
```

```
if (un.item.sizeArr[0] == 10 || un.item.sizeArr[1] == 10 || un.item.sizeArr[2]  
== 10){  
    printf("Input data is consisted a forbidden number (10), so the program  
ends.");  
    return ERROR_INPUT_DATA;  
}
```

```
char * buffer = (char *)(malloc(maxsize * sizeof(char)));  
encBuffer(un.arrB, buffer, maxsize, encrypt);
```

```
FILE *f = fopen(argv[1], "wb");  
fputs(buffer, f);
```

```
    free(buffer);  
    fclose(f);  
    return 0;  
}
```

ПРИЛОЖЕНИЕ 2

```
#include <stdio.h>
#include <malloc.h>
#include <stdlib.h>
#define MAX_NAME_SIZE 1000
#define QUANTITY 3
#define ERROR_OPEN_FILE -3
#define ERROR_INPUT_DATA -1
```

```
struct Item{
    int sizeArr[QUANTITY];
    char name[MAX_NAME_SIZE];
};
```

```
union U{
    struct Item item;
    char arrB[sizeof(struct Item)];
};
```

```
char decrypt(char byte){
    byte += 10;
    return byte;
}
```



```

void decBuffer(const char * inBuffer, char * outBuffer, unsigned int len, char
(*dPtr)(char )){
    for (int i = 0; i < len; i++){
        *(outBuffer + i) = dPtr(*(inBuffer + i));
    }
}

```

```

int main(int argc, char ** argv){

    if (argc != 2){
        printf("Expected two arguments: the running file and the file to write from
a data");
        return ERROR_INPUT_DATA;
    }

```

```

    union U un;
    int maxsize = sizeof(struct Item);
    char * buffer = (char *)malloc(maxsize * sizeof(char));
    FILE *f = fopen(argv[1], "rb");
    if (f == NULL) {
        printf("Error opening file");
        free(buffer);
        return ERROR_OPEN_FILE;
    }
    fgets(buffer, maxsize + 1, f);
    decBuffer(buffer, un.arrB, maxsize, decrypt);
    free(buffer);
    fclose(f);

```

```
printf("%s\n", un.item.name);  
for (int i = 0; i < QUANTITY; i++){  
    printf("%d. %d\n", i + 1, un.item.sizeArr[i]);  
}  
return 0;  
}
```