Лабораторная работа №3 Асылбек уулу Бакыта из группы ИСП-308

Работа с файлами

1 часть

1. Удалить файл, имя которого задано в командной строке.

#include < stdio.h >

#include <Windows.h>

int main(int argc, char\* args[])

{

printf("Delete this file: ");

if (!DeleteFileA(args[1])) printf("\n error");

else

{

if (args[1] == NULL) printf("error\n");

else

{

if (argc != 2) printf("error\n");

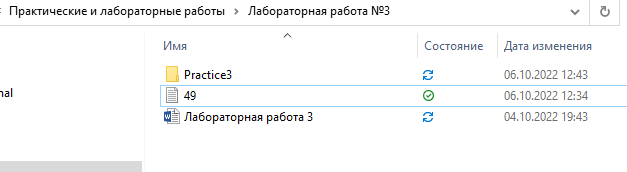
else printf("file was deleted");

}

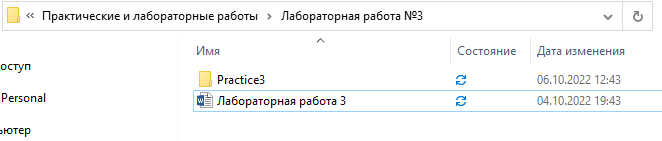
}

return 0;

}







1. Переименовать файл, старое и новое имя задаются в командной строке.

#include < stdio.h >

#include <Windows.h>

int main(int argc, char\* args[])

{

printf("Renaming a file: ");

if (!MoveFileA(args[1], args[2])) printf("\nerror");

else

{

if ((args[1] == NULL) && (args[2] == NULL)) printf("error\n");

else

{

if (argc != 3) printf("error\n");

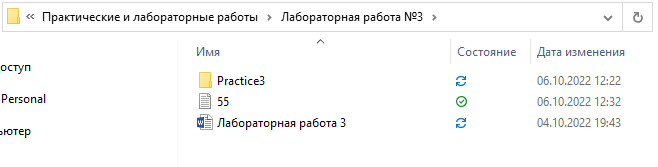
else printf("\nthe file has been renamed");

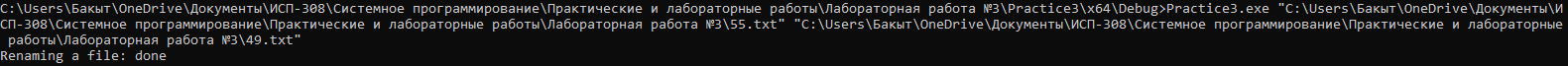
}

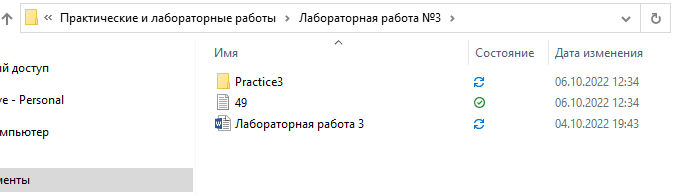
}

return 0;

}







1. Копировать файл из одной папки в другую. Имена файла и папок задаются в командной строке.

#include <stdio.h>

#include <Windows.h>

int main(int argc, char\* args[])

{

printf("Copying a file");

if (!CopyFileA(args[1], args[2], FALSE)) printf("\nThe file was not copied");

else

{

if ((args[1] == NULL) && (args[2] == NULL)) printf("error\n");

else

{

if (argc != 3) printf("error\n");

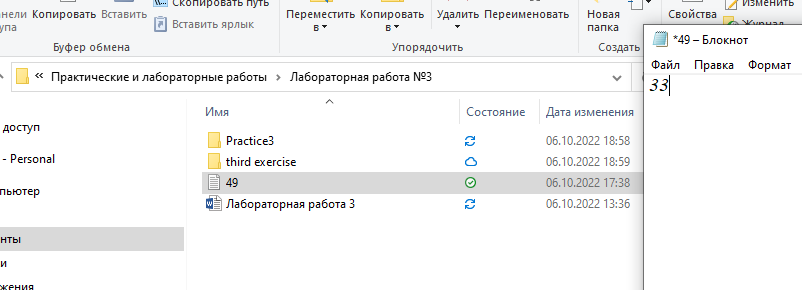
else printf("\nThe file has been copied");

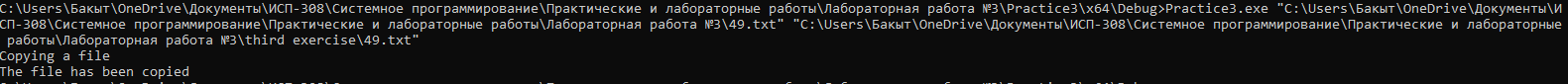
}

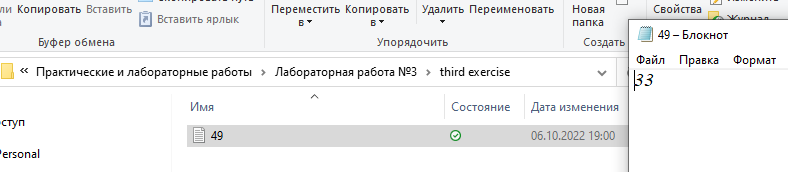
}

return 0;

}







1. Вывести на экран размер файла, имя которого задано в командной строке.

#include < stdio.h >

#include <Windows.h>

int main(int argc, char\* args[])

{

HANDLE file = INVALID\_HANDLE\_VALUE;

file = CreateFileA(args[1], FILE\_GENERIC\_READ | FILE\_GENERIC\_WRITE, FILE\_SHARE\_READ | FILE\_SHARE\_WRITE, 0, OPEN\_ALWAYS, FILE\_ATTRIBUTE\_NORMAL, NULL);

if (file == INVALID\_HANDLE\_VALUE)

{

printf("CreateFile failed\n");

}

if (!file) printf("\nThe file was not created");

else

{

if (args[1] == NULL) printf("error\n");

else

{

if (argc != 2) printf("error\n");

else

{

printf("\nThe file has been created");

DWORD f = GetFileSize(file, 0);

if (f == INVALID\_FILE\_SIZE)

{

printf("GetFileSize failed\n");

CloseHandle(file);

}

printf("file size %d", f);

}

}

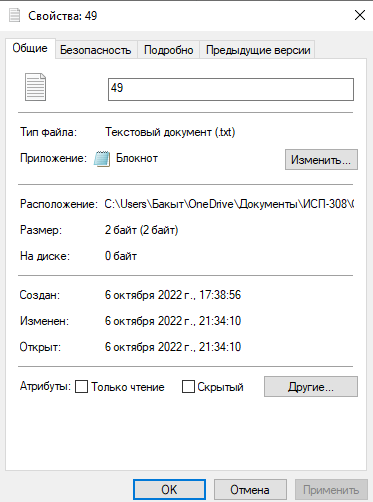
}

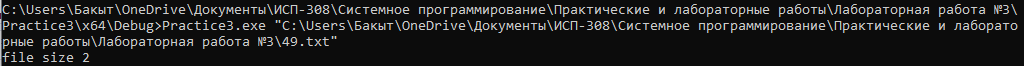
if (!CloseHandle(file)) printf("\nerror");

else printf("\nThe file is closed");

return 0;

}





1. Вывести на экран атрибуты файла, имя которого задано в командной строке.

#include < stdio.h >

#include <windows.h>

int main(int argc, char\* args[])

{

WIN32\_FILE\_ATTRIBUTE\_DATA buf = { 0 };

if (!GetFileAttributesExA(args[1], GetFileExInfoStandard, &buf)) printf("\nerror");

else

{

if (args[1] == NULL) printf("error\n");

else

{

if (argc != 2) printf("error\n");

else

{

if (buf.dwFileAttributes & FILE\_ATTRIBUTE\_ARCHIVE) printf("The file has archive attribute\n");

else printf("error\n");

if (buf.dwFileAttributes & FILE\_ATTRIBUTE\_HIDDEN) printf("The file has hidden attribute\n");

else printf("error\n");

if (buf.dwFileAttributes & FILE\_ATTRIBUTE\_NORMAL) printf("The file has normal attribute\n");

else printf("error\n");

if (buf.dwFileAttributes & FILE\_ATTRIBUTE\_READONLY) printf("The file has read-only attribute\n");

else printf("error\n");

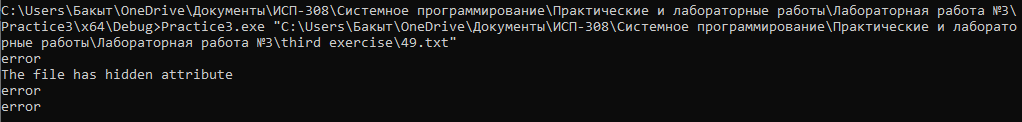
}

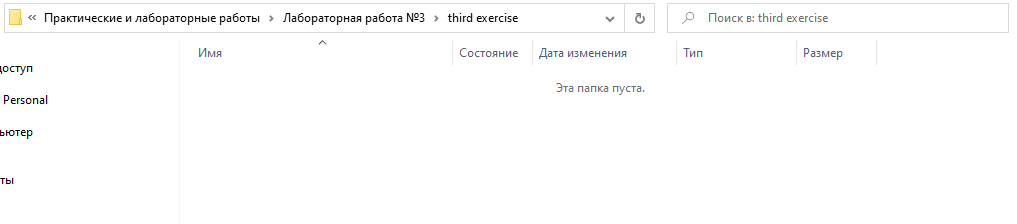
}

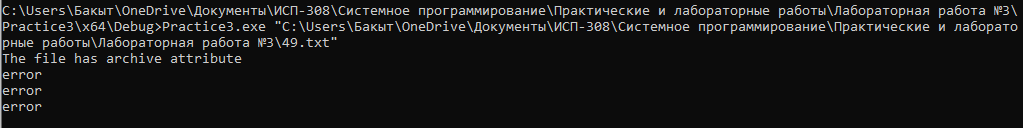
}

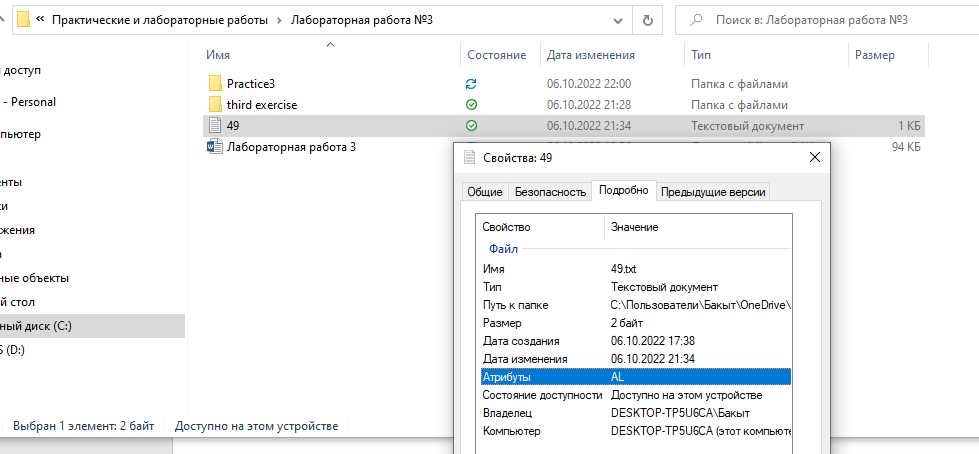
return 0;

}









1. Установить атрибут «только для чтения» файлу, имя которого задано в командной строке.

#include < stdio.h >

#include <windows.h>

int main(int argc, char\* args[])

{

if (!SetFileAttributesA(args[1], FILE\_ATTRIBUTE\_READONLY)) printf("\nerror");

else

{

if (args[1] == NULL) printf("error\n");

else

{

if (argc != 2) printf("error\n");

else

{

WIN32\_FILE\_ATTRIBUTE\_DATA buf = { 0 };

GetFileAttributesExA(args[1], GetFileExInfoStandard, &buf);

if (buf.dwFileAttributes & FILE\_ATTRIBUTE\_READONLY) printf("The read-only attribute is set");

else printf("error");

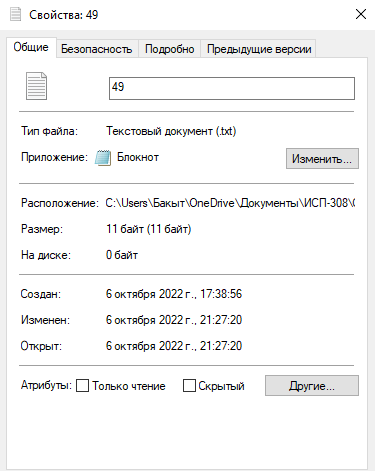
}

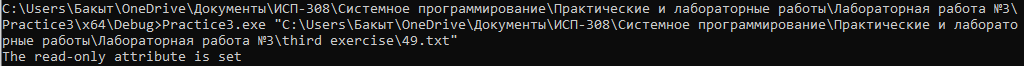
}

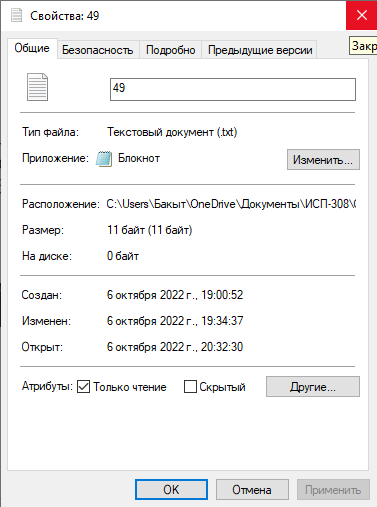
}

return 0;

}







1. Установить атрибут «скрытый» файлу, имя которого задано в командной строке.

#include < stdio.h >

#include <windows.h>

int main(int argc, char\* args[])

{

if (!SetFileAttributesA(args[1], FILE\_ATTRIBUTE\_HIDDEN)) printf("\nerror");

else

{

if (args[1] == NULL) printf("error\n");

else

{

if (argc != 2) printf("error\n");

else

{

WIN32\_FILE\_ATTRIBUTE\_DATA buf = { 0 };

GetFileAttributesExA(args[1], GetFileExInfoStandard, &buf);

if (buf.dwFileAttributes & FILE\_ATTRIBUTE\_HIDDEN) printf("The hidden attribute is set");

else printf("error");

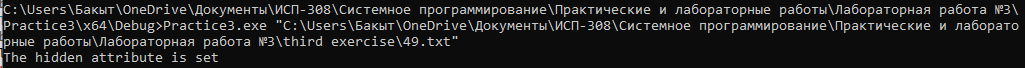
}

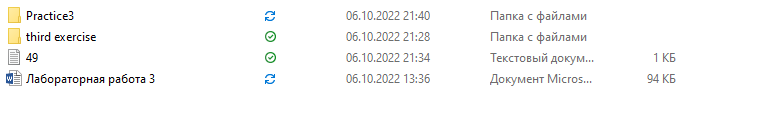
}

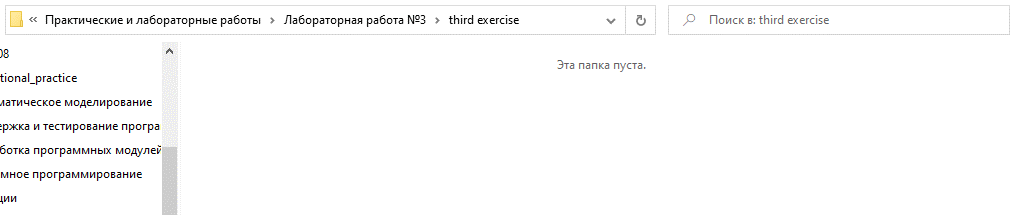
}

return 0;

}







Я забыл сделать скрин до выполнения программы. И теперь я не знаю, как показать и удалить тот файл.

1. Вывести на экран содержимое папки, имя которой задано в командной строке. Выводить только имена (и типы) файлов и папок.

#include <windows.h>

#include <tchar.h>

#include <stdio.h>

#include <strsafe.h>

void \_tmain(int argc, TCHAR\* argv[])

{

WIN32\_FIND\_DATA f;

LARGE\_INTEGER filesize;

TCHAR dir[MAX\_PATH];

HANDLE hf = INVALID\_HANDLE\_VALUE;

// Prepare string for use with FindFile functions. First, copy the string to a buffer, then append '\\*' to the directory name.

if (!StringCchCopy(dir, MAX\_PATH, argv[1])) printf("\nerror");

else

{

if (argv[1] == NULL) printf("error\n");

else

{

if (argc != 2) printf("error\n");

else

{

printf("the folder contains: ");

}

}

}

StringCchCat(dir, MAX\_PATH, TEXT("\\\*"));

// Find the first file in the directory.

hf = FindFirstFile(dir, &f);

// List all the files in the directory with some info about them.

do

{

if (f.dwFileAttributes & FILE\_ATTRIBUTE\_DIRECTORY)

{

\_tprintf(TEXT(" %s <DIR>\n"), f.cFileName);

}

else

{

filesize.HighPart = f.nFileSizeHigh;

\_tprintf(TEXT(" %s\n"), f.cFileName);

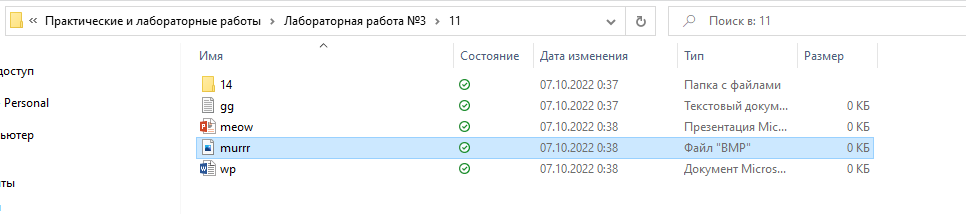
}

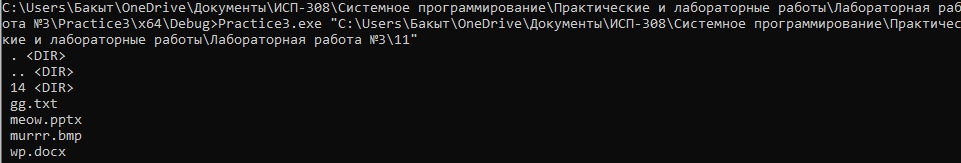
} while (FindNextFile(hf, &f) != 0);

if (!FindClose(hf)) printf("\nerror");

else printf("\nThe file is closed");

}





1. Создать заданную папку в текущей. Имя папки задается в командной строке.

#include < stdio.h >

#include <Windows.h>

int main(int argc, char\* args[])

{

printf("create a folder\n");

if (!CreateDirectoryA(args[1], 0)) printf("error\n");

else

{

if (args[1] == NULL) printf("error\n");

else

{

if (argc != 2) printf("error\n");

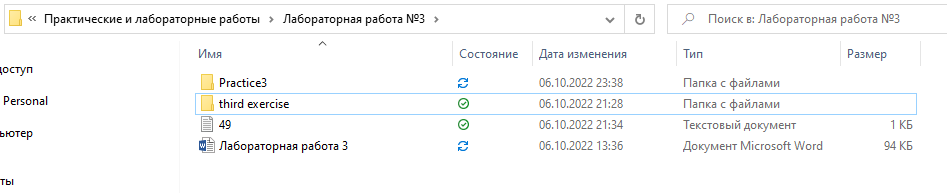
else printf("the folder was created\n");

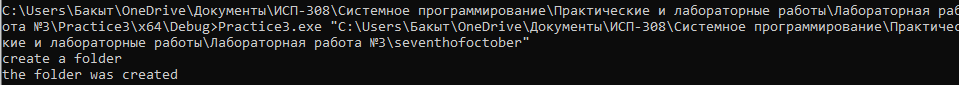
}

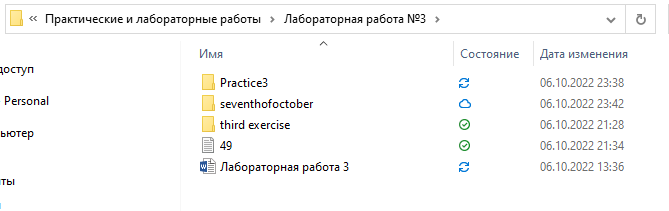
}

return 0;

}







1. Переместить файл из одной папки в другую. Имена файла и папок задаются в командной строке.

#include < stdio.h >

#include <Windows.h>

int main(int argc, char\* args[])

{

printf("moving a file:\n");

if (!MoveFileA(args[1], args[2])) printf("error\n");

else

{

if ((args[1] == NULL) && (args[2] == NULL)) printf("error\n");

else

{

if (argc != 3) printf("error\n");

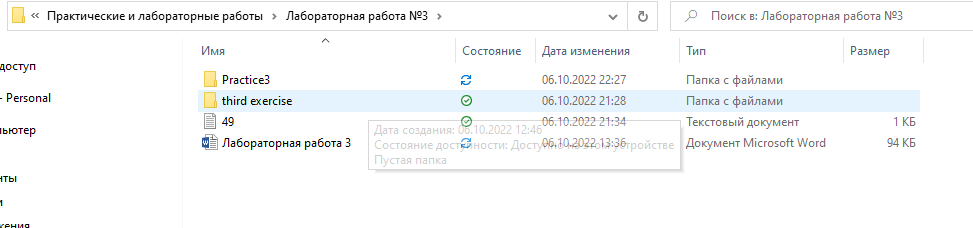
else printf("the file has been moved\n");

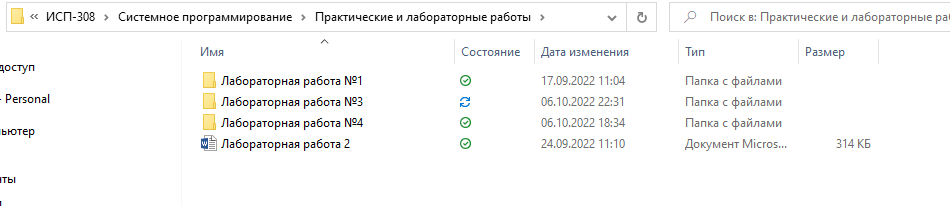
}

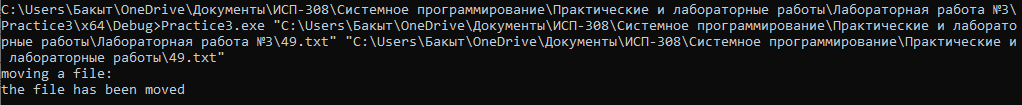
}

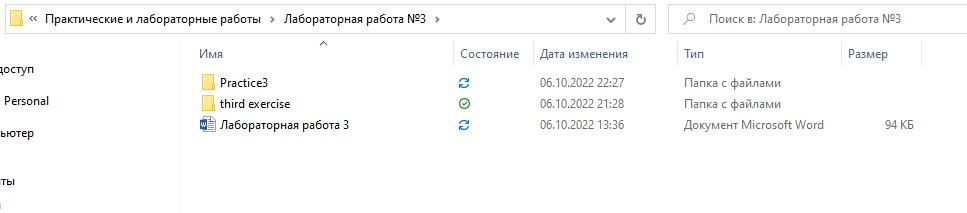
return 0;

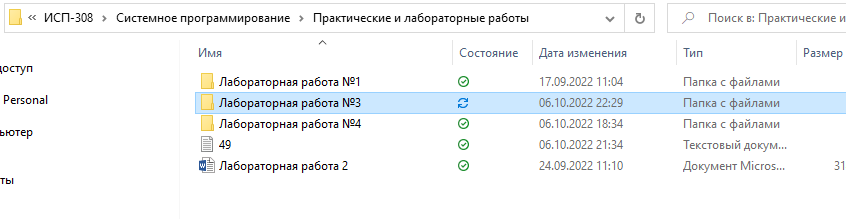
}











2 часть

1. Отсортировать по убыванию содержимое числового файла

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <windows.h>

#include <stdio.h>

#include <string.h>

#include < stdlib.h >

int main(int argc, char\* args[])

{

HANDLE f = CreateFileA(args[1], GENERIC\_WRITE | GENERIC\_READ, FILE\_SHARE\_READ | FILE\_SHARE\_WRITE, 0, OPEN\_EXISTING, FILE\_ATTRIBUTE\_NORMAL, 0);

if (f == INVALID\_HANDLE\_VALUE)

{

printf("CreateFile failed\n");

}

DWORD fs = GetFileSize(f, NULL);

if (fs == INVALID\_FILE\_SIZE)

{

printf("GetFileSize failed\n");

CloseHandle(f);

}

HANDLE mapf = CreateFileMappingA(f, NULL, PAGE\_READWRITE, 0, 0, NULL);

if (mapf == NULL)

{

printf("CreateFileMapping failed\n");

CloseHandle(f);

}

LPVOID address = MapViewOfFile(mapf, FILE\_MAP\_ALL\_ACCESS, 0, 0, 0);

if (address == NULL)

{

printf("MapViewOfFile failed\n");

CloseHandle(mapf);

CloseHandle(f);

}

char\* string = (char\*)address;

printf("%s\n", string);

int size = strlen(string);

for (int i = 0; i < size; i++)

{

for (int j = 0; j < size - 1; j++)

{

if (string[j] < string[j + 1])

{

int b = string[j];

string[j] = string[j + 1];

string[j + 1] = b;

}

}

}

printf("\n%s", string);

if (!UnmapViewOfFile(address)) printf("\nerror");

else printf("\nThe function has been executed");

if (!CloseHandle(mapf)) printf("\nerror");

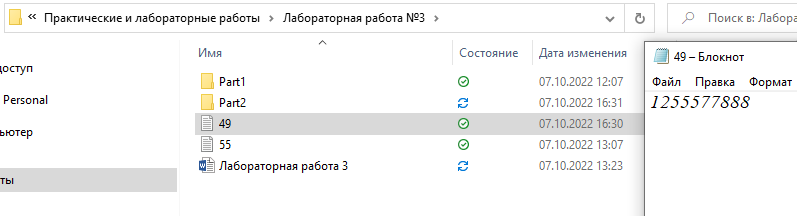
else printf("\nfile mapping closed successfully");

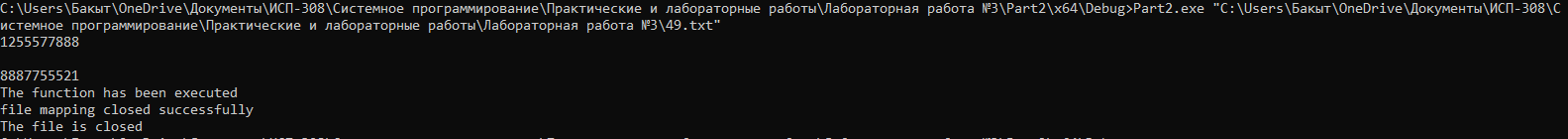
if (!CloseHandle(f)) printf("\nerror");

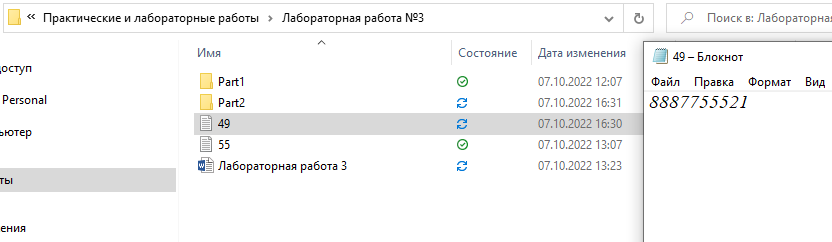
else printf("\nThe file is closed");

return 0;

}







1. Найти максимальное число в файле, содержащем числа

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <windows.h>

#include <stdio.h>

int main(int argc, char\* args[])

{

HANDLE f = CreateFileA(args[1], GENERIC\_WRITE | GENERIC\_READ, FILE\_SHARE\_READ | FILE\_SHARE\_WRITE, 0, OPEN\_EXISTING, FILE\_ATTRIBUTE\_NORMAL, 0);

if (f == INVALID\_HANDLE\_VALUE)

{

printf("CreateFile failed\n");

}

HANDLE mapf = CreateFileMappingA(f, NULL, PAGE\_READWRITE, 0, 0, NULL);

if (mapf == NULL)

{

printf("CreateFileMapping failed\n");

CloseHandle(f);

}

LPVOID address = MapViewOfFile(mapf, FILE\_MAP\_ALL\_ACCESS, 0, 0, GetFileSize(f, 0));

if (address == NULL)

{

printf("MapViewOfFile failed\n");

CloseHandle(mapf);

CloseHandle(f);

}

char\* string = (char\*)address;

printf("\n%s\n", string);

int\* mas = (int\*)malloc(sizeof(int));

int count = 0;

int cpace\_idx = 0;

const int size = strlen(string);

for (int g = 0; g < size; g++)

{

if (string[g] == ' ')

{

count++;

char temp\_string[10] = { 0 };

strncpy(temp\_string, &string[cpace\_idx], g - cpace\_idx);

mas = (int\*)realloc(mas, sizeof(int) \* count);

mas[count - 1] = atoi(temp\_string);

cpace\_idx = g + 1;

}

}

for (int i = 0; i < count; i++)

{

for (int j = i; j < count; j++)

{

if (mas[i] > mas[j])

{

int temp = mas[i];

mas[i] = mas[j];

mas[j] = temp;

}

}

}

printf\_s("max = %d\n", mas[count - 1]);

if (!UnmapViewOfFile(address)) printf("\nerror");

else printf("\nThe function has been executed");

if (!CloseHandle(mapf)) printf("\nerror");

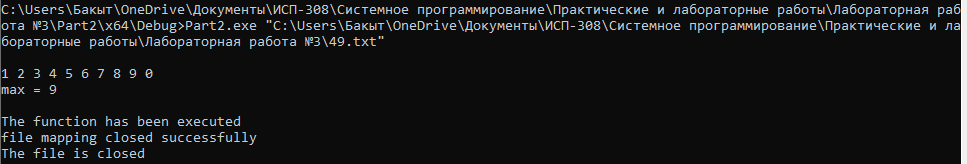
else printf("\nfile mapping closed successfully");

if (!CloseHandle(f)) printf("\nerror");

else printf("\nThe file is closed");

return 0;

}



1. Упорядочить буквы в обратном алфавитном порядке

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <windows.h>

#include <stdio.h>

int main(int argc, char\* args[])

{

HANDLE f = CreateFileA(args[1], GENERIC\_WRITE | GENERIC\_READ, FILE\_SHARE\_READ | FILE\_SHARE\_WRITE, 0, OPEN\_EXISTING, FILE\_ATTRIBUTE\_NORMAL, 0);

if (f == INVALID\_HANDLE\_VALUE)

{

printf("CreateFile failed\n");

}

DWORD fs = GetFileSize(f, NULL);

if (fs == INVALID\_FILE\_SIZE)

{

printf("GetFileSize failed\n");

CloseHandle(f);

}

HANDLE mapf = CreateFileMappingA(f, NULL, PAGE\_READWRITE, 0, 0, NULL);

if (mapf == NULL)

{

printf("CreateFileMapping failed\n");

CloseHandle(f);

}

LPVOID address = MapViewOfFile(mapf, FILE\_MAP\_ALL\_ACCESS, 0, 0, 0);

if (address == NULL)

{

printf("MapViewOfFile failed\n");

CloseHandle(mapf);

CloseHandle(f);

}

char\* string = (char\*)address;

printf("%s\n", string);

int n = strlen(string);

char l = (char)malloc(sizeof(char));

printf("\nsymbols is sorted in descending order\n");

for(int i = 0; i < n; i++)

{

for (int z = 0; z < n; z++)

{

if (string[i] > string[z])

{

l = string[i];

string[i] = string[z];

string[z] = l;

}

}

}

printf("%s", string);

if (!UnmapViewOfFile(address)) printf("\nerror");

else printf("\nThe function has been executed");

if (!CloseHandle(mapf)) printf("\nerror");

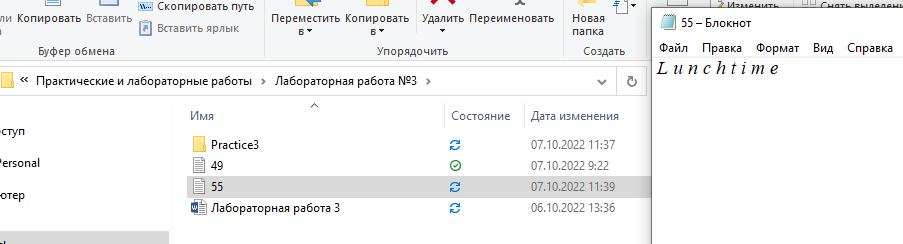
else printf("\nfile mapping closed successfully");

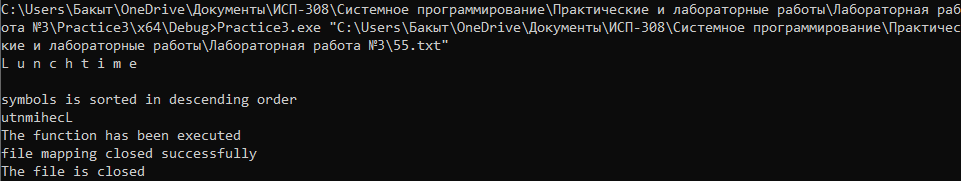
if (!CloseHandle(f)) printf("\nerror");

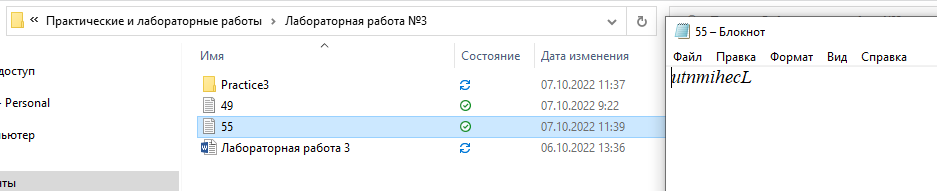
else printf("\nThe file is closed");

return 0;

}







1. Посчитать количество цифр в тексте, содержащем буквенно-цифровую информацию

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <windows.h>

#include <stdio.h>

int main(int argc, char\* args[])

{

HANDLE f = CreateFileA(args[1], GENERIC\_WRITE | GENERIC\_READ, FILE\_SHARE\_READ | FILE\_SHARE\_WRITE, 0, OPEN\_EXISTING, FILE\_ATTRIBUTE\_NORMAL, 0);

if (f == INVALID\_HANDLE\_VALUE)

{

printf("CreateFile failed\n");

}

DWORD fs = GetFileSize(f, NULL);

if (fs == INVALID\_FILE\_SIZE)

{

printf("GetFileSize failed\n");

CloseHandle(f);

}

HANDLE mapf = CreateFileMappingA(f, NULL, PAGE\_READWRITE, 0, 0, NULL);

if (mapf == NULL)

{

printf("CreateFileMapping failed\n");

CloseHandle(f);

}

LPVOID address = MapViewOfFile(mapf, FILE\_MAP\_ALL\_ACCESS, 0, 0, 0);

if (address == NULL)

{

printf("MapViewOfFile failed\n");

CloseHandle(mapf);

CloseHandle(f);

}

char numbers[11] = "1234567890";

char\* string = (char\*)address;

printf("%s\n", string);

int k = 0, n = strlen(string);

for (int i = 0; i < n; i++)

{

for (int j = 0; j < 10; j++)

{

if (string[i] == numbers[j])

{

k++;

}

}

}

printf("\n%i", k);

if (!UnmapViewOfFile(address)) printf("\nerror");

else printf("\nThe function has been executed");

if (!CloseHandle(mapf)) printf("\nerror");

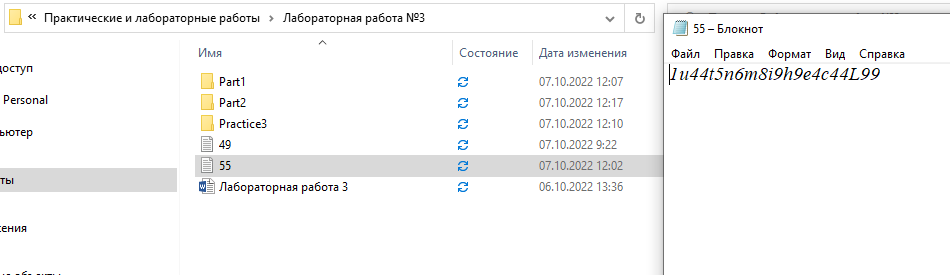
else printf("\nfile mapping closed successfully");

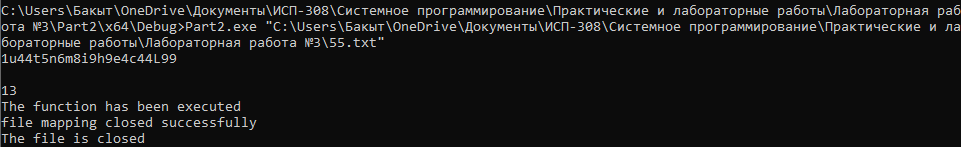
if (!CloseHandle(f)) printf("\nerror");

else printf("\nThe file is closed");

return 0;

}





1. Удалить все буквы «а» из текста. В конец файла записать, сколько букв было удалено.

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <windows.h>

#include <stdio.h>

void a(char\* s)

{

int i, j;

char symbol[3] = "Aa";

for (i = 0; i < strlen(s);i++)

{

for (int z = 0; z < 2; z++)

{

if (s[i] == symbol[z])

{

do {

for (j = i;j < symbol[z];j++)

{

s[j] = s[j + 1];

}

} while (s[i] == symbol[z]);

s[j] = '\0';

}

}

}

}

int main(int argc, char\* args[])

{

HANDLE f = CreateFileA(args[1], GENERIC\_WRITE | GENERIC\_READ, FILE\_SHARE\_READ | FILE\_SHARE\_WRITE, 0, OPEN\_EXISTING, FILE\_ATTRIBUTE\_NORMAL, 0);

if (f == INVALID\_HANDLE\_VALUE)

{

printf("CreateFile failed\n");

}

DWORD fs = GetFileSize(f, NULL);

if (fs == INVALID\_FILE\_SIZE)

{

printf("GetFileSize failed\n");

CloseHandle(f);

}

HANDLE mapf = CreateFileMappingA(f, NULL, PAGE\_READWRITE, 0, 0, NULL);

if (mapf == NULL)

{

printf("CreateFileMapping failed\n");

CloseHandle(f);

}

LPVOID address = MapViewOfFile(mapf, FILE\_MAP\_ALL\_ACCESS, 0, 0, 0);

if (address == NULL)

{

printf("MapViewOfFile failed\n");

CloseHandle(mapf);

CloseHandle(f);

}

char\* string = (char\*)address;

printf("%s\n", string);

int k = 0, n = strlen(string);

char symbol[3] = "Aa";

for (int i = 0; i < n; i++)

{

for (int j = 0; j < 2; j++)

{

if (string[i] == symbol[j])

{

k++;

}

}

}

a(string);

printf("\n %i\n %s", k, string);

if (!UnmapViewOfFile(address)) printf("\nerror");

else printf("\nThe function has been executed");

if (!CloseHandle(mapf)) printf("\nerror");

else printf("\nfile mapping closed successfully");

if (!CloseHandle(f)) printf("\nerror");

else printf("\nThe file is closed");

return 0;

}

